

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



**FEUP**

# **Project and Development of a Case-Based Reasoning Poker Bot**

**Vítor Hugo da Silva Pereira**

Report of Dissertation

Master in Informatics and Computing Engineering

Supervisor: Luis Paulo Reis

March, 2010



# **Project and Development of a Case-Based Reasoning Poker Bot**

**Vítor Hugo da Silva Pereira**

Report of Dissertation  
Master in Informatics and Computing Engineering

Approved in oral examination by the committee:

President: António Augusto de Sousa

---

External Examiner: José Manuel Torres

Internal Examiner: Luis Paulo Reis

23<sup>st</sup> March, 2010



# Abstract

Game research in Artificial Intelligence is a common subject and has been the theme of a wide range of academic works. In order to prove theoretical principals to defeat the human chess champion many games were studied and many scientific approaches were taken in the past. Poker started to appear as an interesting research subject for academic work soon after the first online games gained a broader participation.

Poker agent development is a very challenging domain for using artificial intelligence techniques adapted to a competitive environment with the elements of chance and hidden information. This challenge has been met before with different techniques and different results. A brief review of common approaches to poker play was undertaken together with a deeper review on Case-Based Reasoning approaches.

This dissertation work seeks to provide a framework for agents with online Poker playing capacity and case-based reasoning features. For this purpose PBot, a Poker Bot, was developed and its implementation is documented.

The poker agent developed showed capable to play poker online and successfully used case-based reasoning in its decision-making process. The agent developed presents an opportunity for future improvement and is a first step in building a full-fledge poker agent.



# Resumo

A investigação da temática dos jogos na Inteligência Artificial é comum e tem dado origem a um variado conjunto de trabalhos académicos. Com o objectivo de realizar a prova de princípios teóricos até à vitória contra campeão de xadrez muitos jogos foram estudados e variadas abordagens científicas foram tomados no passado. O Poker, por sua vez, começou a aparecer como um assunto interessante para trabalhos académicos logo que as primeiras salas de jogo online de poker ganharam uma audiência alargada.

O desenvolvimento de agentes de póquer é um domínio muito desafiante para utilizar técnicas de inteligência artificial adaptadas a um ambiente competitivo com elementos aleatórios e de informação oculta. Este desafio foi atingido anteriormente com diferentes técnicas e diferentes resultados. Uma breve análise das várias abordagens ao póquer é feita junto com uma análise mais aprofundada de abordagens de raciocínio baseado em casos.

O trabalho desta dissertação procura providenciar uma estrutura de trabalho para agentes com capacidade de jogar online e funcionalidades de raciocínio baseado em casos. Para este propósito o PBot, um Poker Bot, foi desenvolvido e a sua implementação é documentada.

O agente desenvolvido mostra capacidades de jogar online e utilizar com sucesso raciocínio baseado em casos no seu processo de decisão. O agente desenvolvido expõe uma oportunidade para melhoria futura e é um primeiro passo para um agente de póquer vencedor.





# Acknowledgements

I would like to express my gratitude to:

Luis Paulo Reis for the dissertation orientation and the continuous advisement

My family and friends for their support and interest in this work

Microsoft for providing without any costs the very useful tools Microsoft Visual Studio and Microsoft .Net framework

All contributors and supporters of the following, and related, open source projects that allowed me to complete this work: OpenHoldem, PostgreSQL, Notepad++, MiKTeX, DokuWiki and PokerTH.

Vítor Hugo da Silva Pereira



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	2
1.2	Goals . . . . .	2
1.3	Document structure . . . . .	2
<b>2</b>	<b>Poker</b>	<b>5</b>
2.1	Texas Hold'em . . . . .	5
2.2	Bet Limits . . . . .	8
2.2.1	Fixed Limit . . . . .	9
2.2.2	Pot Limit . . . . .	9
2.2.3	No Limit . . . . .	9
2.3	Basic strategy . . . . .	10
2.4	Playing Sites . . . . .	11
2.5	Conclusions . . . . .	12
<b>3</b>	<b>State of Art</b>	<b>13</b>
3.1	Approaches To Computer Poker . . . . .	13
3.1.1	Rule-based . . . . .	14
3.1.2	Simulation . . . . .	14
3.1.3	Game theory . . . . .	15
3.1.4	Adaptive strategies . . . . .	16
3.1.5	Case-based Reasoning . . . . .	16
3.2	Online Poker . . . . .	18
3.2.1	IRC games . . . . .	18
3.2.2	Online Poker Sites . . . . .	18
3.2.3	PokerTH . . . . .	19
3.3	Conclusions . . . . .	20
<b>4</b>	<b>PBot: A Case-Based Resoning Poker Bot</b>	<b>21</b>
4.1	Solution Architecture . . . . .	21
4.2	Online play . . . . .	23
4.2.1	Perception emulation . . . . .	23
4.2.2	OpenHoldem . . . . .	25
4.2.3	User DLL . . . . .	29
4.3	PBot reason engine . . . . .	32
4.3.1	Communication . . . . .	33
4.3.2	PBot . . . . .	34

## CONTENTS

4.3.3	Case specification . . . . .	36
4.3.4	Case Recall . . . . .	38
4.3.5	Case Reuse . . . . .	38
4.3.6	Case Revise . . . . .	40
4.3.7	Case store . . . . .	40
4.4	Conclusions . . . . .	40
<b>5</b>	<b>Results and Tests</b>	<b>43</b>
5.1	Table Scrap . . . . .	43
5.2	OpenHoldem & PBot interaction . . . . .	45
5.3	PostgreSQL integration . . . . .	45
5.4	Poker play . . . . .	46
5.5	Conclusions . . . . .	46
<b>6</b>	<b>Conclusions and Future Work</b>	<b>47</b>
6.1	Goal achievements & Future Work . . . . .	47
	<b>References</b>	<b>49</b>
<b>A</b>	<b>Glossary of Poker Terms</b>	<b>53</b>
<b>B</b>	<b>169 Hole Cards Rank</b>	<b>57</b>

# List of Figures

2.1	Poker variation schematization (taken from [Que]) . . . . .	6
2.2	Poker hands combination examples . . . . .	7
2.3	Poker showdown example . . . . .	8
2.4	Poker showdown hands . . . . .	9
2.5	A Poker table representation with the blinds bet . . . . .	10
2.6	PartyPoker Poker revenues . . . . .	12
3.1	Homemade Poker bot rules (taken from [Dev]) . . . . .	15
3.2	Case-Based Reasoning cycle. Taken from [dMMB <sup>+</sup> 05]. . . . .	17
3.3	An active IRC Poker network greeting message . . . . .	19
3.4	Full Tilt Poker lobby . . . . .	20
4.1	PBot architecture . . . . .	22
4.2	PokerTH interface, an example of standard poker site software interface . . . . .	24
4.3	Example of a log text box and below the text extracted by the OCR . . . . .	25
4.4	OpenScrape main window with several fields defined . . . . .	26
4.5	Colour hash graphical editor . . . . .	28
4.6	OpenHoldem capturing PokerTH table info . . . . .	29
4.7	struct holdem_state . . . . .	31
4.8	PBot core schematic organization . . . . .	32
4.9	The processing code of the cases . . . . .	39
5.1	OpenHoldem information capture along NoiQPoker table image . . . . .	44
5.2	PBot log . . . . .	45

## LIST OF FIGURES

# List of Tables

4.1	Case-Based Reasoning method association . . . . .	36
4.2	Case fields . . . . .	37
4.3	A simplified example of case recall . . . . .	38

## LIST OF TABLES



# Abbreviations

API	Application Programming Interface
CBR	Case-Based Reasoning
COM	Component Object Model
DBMS	Database Management System
DLL	Dynamic-Link library
IPC	Inter-Process Communication
IRC	Internet Relay Chat
LAN	Local Area Network
OCR	Optical Character Recognition
OH	OpenHoldem
UACPRG	Univesity of Alberta Computer Poker Research Group
WSOP	World Series Of Poker

## ABBREVIATIONS

# Chapter 1

## Introduction

Poker is an interesting game because of the challenge it presents with its elements of luck and hidden information. It may not be the game with the best reputation because its association with money bets and fortune losses but that is only part of what the game involves. Even without the money attraction, Poker is still a very attractive game because of the mixture of luck and skill required to win.

Poker is a century old card game but only recently, in the last decades, its popularity sky rocketed throughout the world. Poker popularity increase began in 1973 with the first television broadcast World Series Of Poker (WSOP) tournament. Walter "Puggy" Pearson won the 1973 tournament for a \$130.000 prize. In the years after, the number of entrants grew gradually from 6 in 1971, the first edition of WSOP, to 839 in 2003. In 2003 the first WSOP winner that qualified from an online tournament won 2,5 million dollars after only paying \$39 to an online satellite tournament entrance. The idea of newcomer challenging the best players to win the first prize gave poker and online poker another popularity boost. In 2004 there were 2576 entrances competing for a \$5,000,000 first prize.

For the academia, Poker was a subject of study early as the first studies of Game Theory. Poker was used as a mean to prove a theory in game theory and not as the subject of study itself. When Poker started to be played online the first attempts to apply Artificial Intelligence ideas to an agent that could play were tested. The first bots were actually made by poker players with programming knowledge and then only the first studies in the academia produce his first's autonomous players.

The challenge was set even before the academic studies were made, how to create a poker player that could beat the stronger human opponents. To this day only part of this challenge was met and this dissertation work follows the path set by that challenge.

## 1.1 Motivation

The main motivation in a poker agent development is to know that, for more than one decade, they have been developed without founding a definitive solution. In contrast to other artificial intelligence games approaches like chess, poker is still an open problem. The confrontation of man vs artificial intelligence is still won by man in almost all poker game forms.

Another motivation for this dissertation is its theme being directly related to such interesting game as poker. It's played around world by millions of people and tournaments with millions of dollars in prizes are annually held. Besides this, poker is a fun game of chance and complex strategy.

## 1.2 Goals

This dissertation main goal is to contribute to the work of others in the agent poker development since it's not expectable that sole dissertation work can outperform the poker agents that have been developed for years.

The principal goal of this dissertation can be set as to the lay of the groundwork to a full-fledge poker agent. This dissertation work seeks to provide the first step into the creation a winning poker agent by the pursuit three objectives:

- **Related work review** — Perform an analysis on common approaches to the poker agent development and the implementations design using case-based reasoning.
- **Online play capacity** — Develop a poker agent capable of play online against human players.
- **Case-Based Reasoning poker bot** — Implement a case-based reasoning methodology process in the poker agent decision making.

## 1.3 Document structure

This document is divided in six chapters.

In the introductory chapter, the first, is described broadly the theme of the dissertation, its motivations and goals, and is given an overview of the document.

The second chapter is a brief description of poker and its characteristics. A general poker description is given as well the Texas Hold'em rules and a brief description of the poker boom history.

The third chapter is a review of the state of art related to the creation of a Poker agent.

In the fourth chapter PBot, the Poker agent, implementation is presented and discussed.

## Introduction

The assessment of the PBot achievements is done the fifth chapter.

Finally, in the last chapter, is exposed an appraisal of this thesis work and suggestions of future work are given.

## Introduction

## Chapter 2

# Poker

Poker is a table card game played with an American deck of 52 cards in which the main objective is to win chips against opponents. Poker can be played in the form of a Poker Tournament or a Ring Game. Both types of poker playing have the same basic principal of being played in successive independent iterations named Hands with the player keeping his chips from the end of one Hand to the start of the next Hand.

A Poker Hand is a sequence of phases in which each phase consist of new cards being dealt and a following betting round. The cards dealt can be player private (Hole Card) or shared (Common Card). The betting round, like in other bet games, only ends when all players even the higher bet amount in successive betting decisions.

The betting decisions consists essentially on choosing between leaving the hand action (Fold) or continue playing committing (Bet) a variable amount of chips. All the committed chips are the prize to the Hand winner, which can be won by one player if all the others players leave the hand action (Fold) or if the player has the strongest combination of cards (Higher Hand Rank), between private and common cards, at the end of the Hand.

The main difference of a Poker Tournament to a Ring Game is the time when a player can enter and leave the competition. In a Poker Tournament the player can only join before the tournament starts and leave if he gets out of chips or has won the tournament. In a Ring Game the poker player can enter or leave when he finds it more favourable, so the overall strategy of a Poker player is different for both types of games.

### 2.1 Texas Hold'em

As Poker has evolved in time and because there are always different ways to play card games a specific type has to be chosen and described to not leave room for confusion. The

## Poker

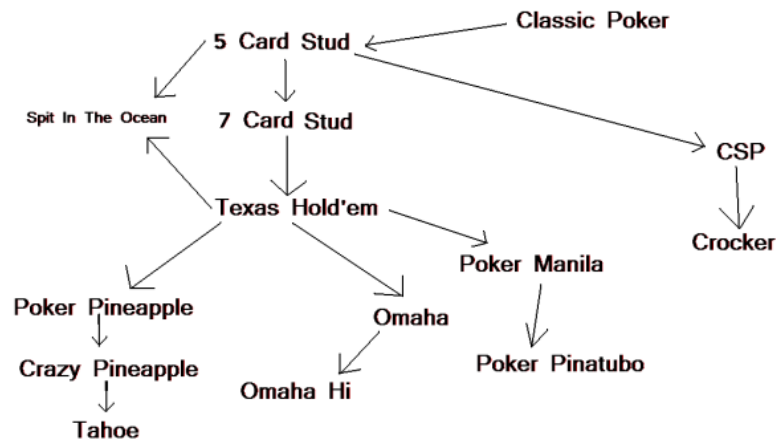


Figure 2.1: Poker variation schematization (taken from [Que])

variation choose in this dissertation is the Texas Hold'em. A schematization of relationship between poker variations and Texas Hold'em can be consulted in the figure 2.1.

Texas Hold'em consists in four phased hand, Pre-Flop, Flop, Turn and River, with a combination of two private cards (Hole Cards) and five common cards. In each hand there is different dealer of the cards (Dealer) which distributes the cards in a clockwise order. The order of the players to take a betting decision is also clockwise.

In the first phase, Pre-Flop, two cards (Hole Cards) are deal in secret to each player in the Table. In this phase, there are two forced bets on the two players on the right of the dealer, called the Blinds. The blinds are of a fixed value before each hand and usually in Ring Games tables have a fixed blind value. The first player on the right of the Dealer pays a Small Blind (half the value of the blind) and next player pays the Big Blind (whole value of the blind). Then the player on the right of the Big Blind, makes his betting decision, then the next and then the remaining, until the betting round is complete.

At the second phase, Flop, three common cards are dealt for everyone to see, and another betting round begins starting with the player on the right of the dealer.

In the third phase, Turn, one common card is dealt and another betting round begins, starting with the player on the right of the dealer.

In the fourth and last phase, River, one common card is dealt with another betting round. In this round, if more than one player is still playing, they end the phase by showing to everyone what they have in their private cards (Hole Cards). When the cards are revealed it can be figured which player has the highest hand. In a case of tie, the pot (all the chips bet) is split between the players.

A player hand is set by the highest card combination possible between the player private cards and the five shared cards. The player can use both, one or none of his private cards for this combination. The classification of hands is done firstly by the following



## Poker

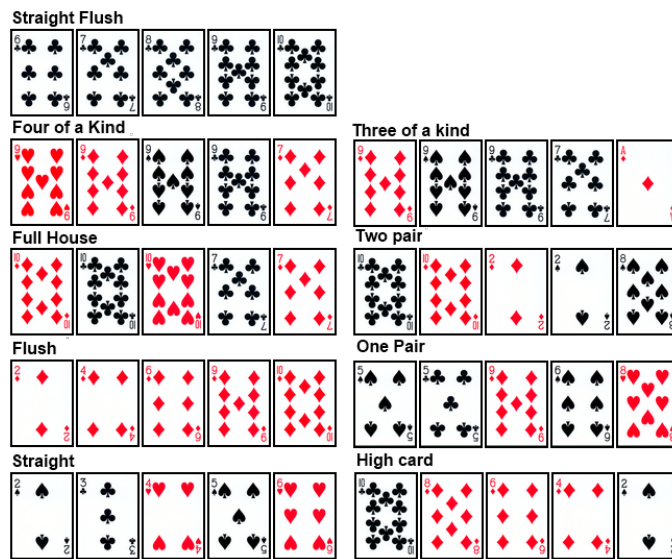


Figure 2.2: Poker hands combination examples

card combinations. An enumeration and description is given below and image examples of the same hands are given in the figure 2.2.

1. **Straight Flush** — Five card sequence of the same suit
2. **Four of a Kind** — Four cards of the same rank and one unrelated card
3. **Full House** — Two cards of the same rank and other three cards of the another common rank
4. **Flush** — Five cards of the same suit
5. **Straight** — Five card sequence of different suits
6. **Three of a kind** — Three cards of the same rank and two unrelated cards
7. **Two pair** — Two cards of the same rank, two other cards of another rank and one unrelated card
8. **One pair** — Two cards of the same rank and three unrelated cards
9. **High card** — The higher card and four unrelated cards

There is a possibility that more than player hold a combination of cards of the same card combination but hands are untie by the cards rank value. First is taken into account the cards rank value relevant to the card combination (e.g. the king in pair a one pair card combination) and then, one by one, from the highest card to lowest, the unrelated cards to the combination are compared. To exemplify this hand classification comparison a hypothetical situation with three players is given.

## Poker

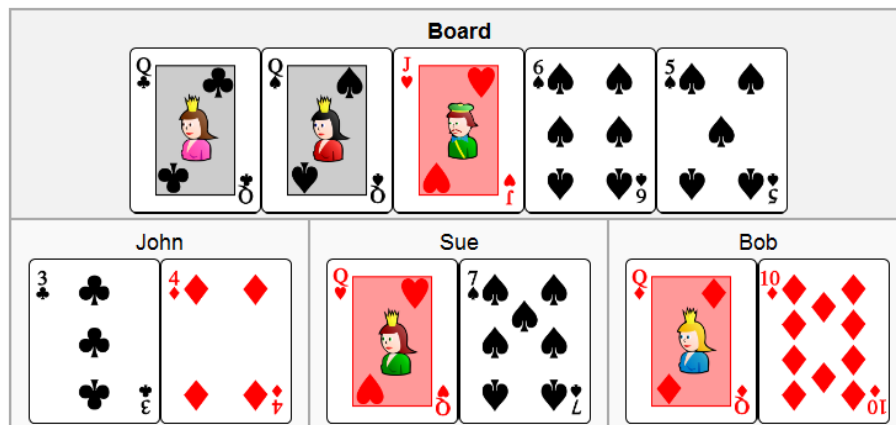


Figure 2.3: Poker showdown example

John, Sue and Bob are in the final phase, the river, and all they go to the shown down, the moment when they show their private cards to evaluate who has the highest hand. John holds a three and a four. Sue holds a queen and a seven. Bob has a queen and a ten. The community cards are two queens, a jack, a six and a five. In the showdown all three players make the best five card combination, so John ends with a hand being composed of only community cards, two queens, a jack, a six and a five because his cards are lower. Sue has the hand with three queens, a jack and seven. Bob's hand is three queens, a jack and a ten.

In the showdown, Bob is the winner. The hand of John is the lowest hand of three with a pair of queens which is inferior to the combination three of a kind of Sue and Bob. Both Sue and Bob have a three of a kind with the same card value, the queen so the other cards are compared. The first higher unrelated card is the jack, but both hold it. The next card of Sue is the seven and in Bob's hand is the ten, and so makes Bob the player with the best hand. This example can be visualized in figure 2.3 and 2.4.

## 2.2 Bet Limits

As mentioned earlier after new cards being dealt the players do their betting in a new betting round. Like in the generic structure of poker there are variations in the betting rules. The three common betting rules are Fixed Limit, Pot Limit and No Limit. No Limit is considered the most popular and the most played in online Poker sites. The Fixed Limit is present almost always but in a small scale and Pot Limit as even less players. All these three types of betting have in common the existence of the big blind and the small blind. The Big Blind is the double of the Small Blind, and their value is fixed. In a Poker Tournament the Blinds vary in time but in Ring Games they are commonly of fixed value per table. A table with blinds bet is shown in figure 2.5.

## Poker

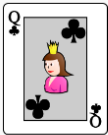
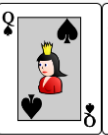



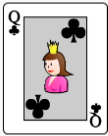

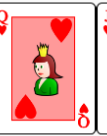


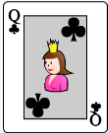

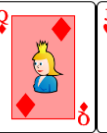


John						A pair of queens
Sue						Three queens, with Jack, Seven kickers
Bob						Three queens, with Jack, Ten kickers

Figure 2.4: Poker showdown hands

### 2.2.1 Fixed Limit

The Fixed Limit betting structure is the more restrictive. It limits the bet amount and the number of bets in each round. The bet amount is of fixed value, the Big Blind, meaning that the only bet that a player can make is with the amount value of the Big Blind. The number of bets to each player in every bet round is limited to four. This consist first of a bet, secondly a raise, thirdly a re-raise, and lastly a cap, a final raise.

### 2.2.2 Pot Limit

Pot limit rules are more permissive than Fixe Limit but still more restrictive than No Limit. In Pot Limit the bet value is limited with a minimum and maximum value. The minimum bet value is as the same size of the Big Blind. The maximum bet size is the equivalent to the size of the table pot. The number of bets isn't limited.

### 2.2.3 No Limit

No Limit is the most popular betting structure and doesn't have any limit in the bet values or in the number of bets. The maximum that a player can bet is limited only by the player own pot size. Because the player pot size can be inferior to bet he may want to call the player can go All-In. All-in is in other words the calling of a bet with less chips he would have by investing the player complete pot. The all-in move allows a player to stay in game risking all his chips but with the possibility to increase his pot greatly. If the player wins, he will not win all the pot but part of the pot relative to his bet amount.

No Limit is the most challenging betting rule because it permits a more diversified play. It allows a player to win increase his chip amount rapidly but at the same time losing it rapidly.

## Poker

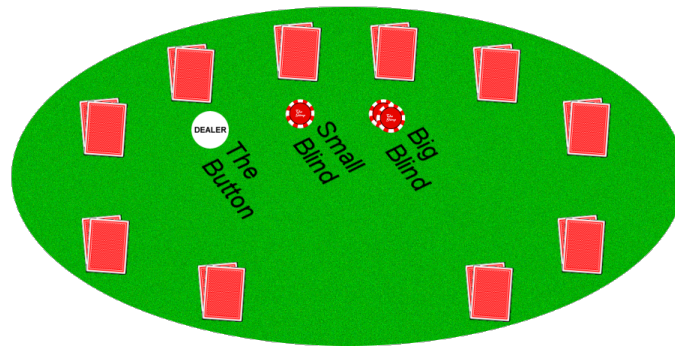


Figure 2.5: A Poker table representation with the blinds bet

### 2.3 Basic strategy

David Sklansky wrote in his influential book 'The Theory of Poker' that "In most games the bets you save are as important as the bets you win, because your real goal is to maximize your wins and minimize your losses" [Sk104]. This is basic guidance of Poker strategy, bet when you win and fold when you lose. Because Poker is a game with incomplete information and stochastic elements a player almost never will be know for sure if he is going to win or lose previously. The only option that a player has is to unbalance to his favour the odds of winning and losing. In other words, a player should seek with his actions to increase the chance of profit and decrease the probability of losses.

In Poker, in similarity to other areas of life, the more a person believes that an investment is safe the more he increases his investment amount in order to benefit from greater profits. In Poker the player investments are his bets and as such the player belief of return of investment will show off most of the times in bets he commits to a hand. Because of this any player can make simple assessments of the other players cards value, so when a player raise (bet more than he is required) he is showing his belief that he has good cards. By contrast, a player that fold or just check (continue in the hand without betting) the player sends a message that he believe that his hand is weak.

Knowing this, many strategies can be derived but a basic strategy is described next. If the player own hand strength is greater than the perceptible hand strength of the opponents the player should bet to maximize profits and fold otherwise to save losses. The challenge, off course, is the accuracy of the evaluation of the player own hand and of the opponents.

A known technique to exploit the incomplete information of the game is to bluff, that is to say, make the opponent believe that he will lose even if he has better cards. This is possible because the other players' perception can be manipulated through betting because a player can make the opponents believe that he has better cards than he really have by betting more than he would normally do with the current cards. If the player succeeds in the manipulation of the opponents' perception he can therefore exploit them

by winning hands with weaker hands, by betting high enough to force the other players to fold.

A common topic in poker strategy and related to what is briefly discussed here is the player position. The player position influences greatly in its decisions because of the knowledge it can access in the time of his decisions. The first player to make a decision in a betting round is the one with less accessible knowledge about the other player's hands because no one has expressed the trust in their cards by betting. In contrast, the last player is the one with best position because it can make more assumptions about the opponent's cards since he already knows what they have bet. For this reason, the last position in a betting round is considered the best.

To conclude this small assessment of Poker strategy, since Poker strategy is much vaster topic, a player should seek to balance evaluations of his own hand against the opponent's hands with all available information to increase his winnings and restrain the losses.

## 2.4 Playing Sites

Poker has grown from a friends' card game to an international sport in the last forty years. This transformed poker from a card game played locally somewhere between a friend house, a saloon or a casino into international TV featured events with lots of money involved. Firstly, and where it evolved, poker was only played with real cards around some table with more formal rules, like in casinos, or more informal rules at someone home. After the first television broadcasted poker tournaments the game increased greatly in popularity but was still limited by the "physical world" rules that the later coming Internet boom came to overcome. Poker rooms were hugely popular principally because it provided a safe place to play, a place to find many opponents and rookie opponents for experienced players to exploit.

With the advent of the Internet it was to be expected that soon or later someone would start playing Poker online and they did. It's reported that 1990's Poker started to be played in IRC networks and there appeared the first Poker bots [Pap98]. The Poker played in IRC networks was essentially for fun and money games weren't the rule. In January the first of 1998 the first online game for money was played [Wik] at Planet Poker site. From that on the Poker online player continues to grow until this day but an event catapulted online poker popularity. In 2003 World Series Of Poker, the most important Poker tournament of the time, was won by "Chris Moneymaker" the first player that qualified from an online Poker room. He had won the tournament entrance by paying online \$39 to another Poker tournament whose prize was the tournament access while the normal entrance for WSOP was \$10000 [Pokb]. The WSOP biggest prize was won by Jamie Gold in 2006 which totalled a 12 million prize, the biggest prize of a sports or television event. The sum of

## Poker

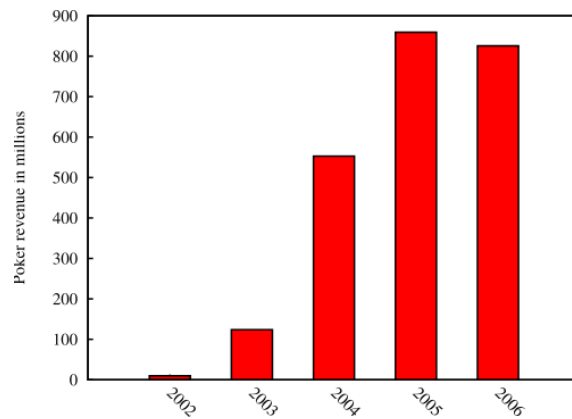


Figure 2.6: PartyPoker Poker revenues

all money prizes of WSOP main event totalled a also incredible 82,5 million dollars. The huge money prizes and TV glamour came to aid poker popularity.

Online Poker sites soon after 2003 marking point became the first place for starters and the place to game. The growth in users and revenues of online poker were hugely boosted in 2004. In 2006 the online Poker players was estimated between from 14 to 23 million players [WW07] and that such user base provided Poker sites an estimated revenue of the \$2,4 billion ( $10^9$ ) in 2005 [Get]. The PartyPoker Poker revenues are shown as an example of the industry growth in the figure 2.6(data from PartyPoker itself).

In a recent Portuguese news report [Ant10], in the SIC television channel, it was reported that online poker sites estimated between 120 to 150 thousands of Portuguese poker players. Around 50 Portuguese players are professional Poker players. Portuguese poker players have also took follow the trend of the international growth of poker.

## 2.5 Conclusions

The description exposed in this chapter depicts the relevant poker involving characteristics, from the basic structure of poker, with its elements of chance and hidden information, to the growth in online poker play. Texas Hold'em, the most popular variation, and the several betting rules associated are descript to allow a better comprehension of the intrinsic specificities of this variations. The basic strategy ground rules in poker play are set to provide a starting point in evaluations taken on the next chapters. Also the description gave on where poker gives the possibility of a better comprehension about the choice in online poker play.

## Chapter 3

# State of Art

In this chapter a review of computer poker is performed. Generic approaches to develop a computer bot are described to provide a comprehension of some possibilities. Then a brief analysis is done on online computer poker. This review will provide the knowledge required to better understanding the options taken in the next chapter.

### 3.1 Approaches To Computer Poker

There are two types of developers in Poker Artificial Intelligence, academic researches and hobbyists. Both types of developers have a different set of goals and produce and publish different types of work. A hobbyist is usually after the challenge of wining money with online Poker and an academic research is more oriented to solve the intellectual challenge of Poker problems and this way improving artificial intelligence capabilities. Besides their different goals, many things differs these two types of developers which demonstrates in the work produce and published.

The homemade Poker bots are usually not very advanced and capable to defeat medium human players. A hobbyist Poker bot can profit, because of play time rewards of some online Poker sites, even with slow pacing loss they are considered good enough with that rate of success. There is no scientific study so far about the prevalence of bots on Poker sites and their capacity of playing so few considerations can be made about them. Now that biggest Poker sites ban users that use bot software and it can be expected that the number of Poker bots active in these sites is low. Also, reading the two biggest Poker bots forums [[Max](#)] [[Poka](#)] there is not much to suspect that homemade Poker bots are more evolved that academic ones. Many homemade Poker bots developers (or hobbyists) actually based their bots on published academic research so they always follow behind. It is important to notice too that homemade bot developers normally share the minimum

amount of information possible in forums and don't follow a tradition of disclosing discoveries so reliable and good information is low.

A review is done on the principal approaches enumerated below and a special emphasis is given to the Case-Based Reasoning.

### 3.1.1 Rule-based

A common description by a Poker player about how he plays consists generally on sentences like, "if I have a pair and the call is low, I call it". This description is a basic rule to play and because it's an intuitive approach Poker bot development has taken it before. These rules try to describe situations and decisions to be taken in those situations. In the past, in the first online games, there were rule based bots that actually were considered good players [Pap98] for bots however these days they are considered not good players [dC09]. This deprecation comes essentially for two reasons, their predictability and the inability to construct rules for covers the most situations possible.

The Poker complexity, calculated in the trillion cases [Bil06], the quantity of rules need to them will surmount to the impossible for a human developer to write. Also there is a problem of quality with the rules because its quality is limited by the poker player that dictates them, in other words, only the best if the best Poker player could enumerate all the possible situations and his best decisions the rules could do almost well as him.

The problem of the predictability is also insurmountable because the rules are static. The problem with predictability is that gives away more information to a observing adversary. If an opponent detects, and players usually seek that, a pattern he can exploit it to its own advantage. The opponent can save losses by inferring that higher cards of the player and manipulate his behavior in his own advantage. For example, a good player knowing that a weak player always folds when the bet is greater than \$20, will always place a bet greater than \$20 to win a pot.

### 3.1.2 Simulation

Enumerating previously all situations for Poker is an impracticable task. However, enumerating all the possible situations, in a given context, might be feasible. This is the principal idea for a simulation approach to Poker playing, take a given game context and evaluate all possible future situations involving the possible own player decisions, the randomness of the cards dealt and the opponents decisions.

The Monte Carlo simulation provides way to effectively perform a simulation relying on repeated computation and pseudo-random inputs. The simulation technique however is not computation feasible with complete random inputs because of its huge computation costs. To deal with this problem not relying pure random inputs are provided but biased



## State of Art

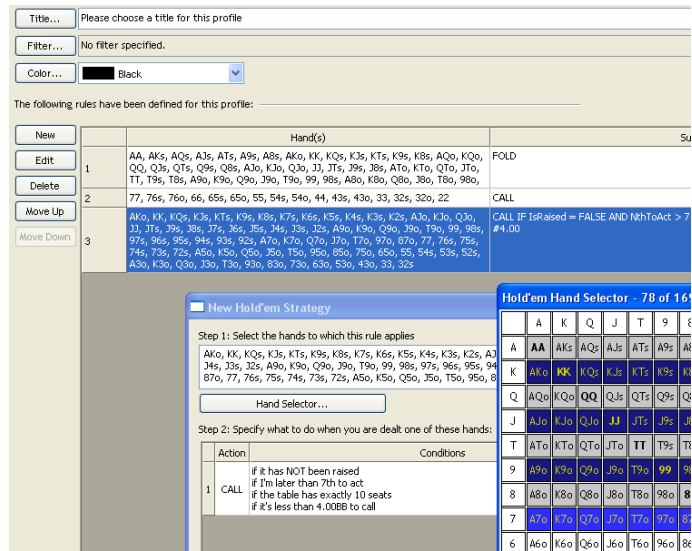


Figure 3.1: Homemade Poker bot rules (taken from [Dev])

inputs. This approach has been proven useful in many other game agents [BPSS99] and also in Poker [Pen99].

### 3.1.3 Game theory

The game theory study gave birth to one of most powerful tools in strategy analysis, Nash Equilibrium and the Equilibrium strategies. The Nash Equilibrium consists in a situation when all agents in a zero-sum competitive environment have no gain from changing their strategy unilaterally. This idea has been used for many studies in economy, sociology and politics between others. The Equilibrium strategy is an attempt to take advantage from Nash Equilibrium abstraction to develop a strategy that can't lose, at least it can tie. An equilibrium strategy may not offer many gains at first sight but the strategy expects the perfect opponent, it expects that the opponent make perfect decisions and since that is not the case in Poker, every time an opponent doesn't make the perfect decision the player has an opportunity to improve his profits.

Poker complexity makes infeasible for in the coming times to encounter an equilibrium strategy, so simplifications are taken. This kind of simplification allowed UACPRG (Univesity of Alberta Computer Poker Research Group) to developed strategies based on Nash Equilibrium that are successful, in particular PsOpti bot series haven be proven strong against very strong human opponents. Never the less, this success is limited to Heads Up games (two player game) [BBD<sup>+</sup>03].

The main disadvantages to Equilibrium strategies are their predictability, their lack of opponent exploiting and being non-adaptive. Because the strategy always expect a perfect opponent even when there is a good opportunity to exploit an opponent weakness the bot

player will not do so. The equilibrium strategy is even so a good strategy when leading with unknown players because it allows detecting patterns in those players without losses. This knowledge about the adversary can be used later to exploit. Notice that the exploit strategy is not an equilibrium strategy.

### 3.1.4 Adaptive strategies

Because of the highly dynamic nature of poker strategies they have to be constantly evaluated and adapted. A good Poker player has to be able to detect the opponents play patterns to take advantage of this information to improve his profits. The adaptation capacity of human and computer poker players rely essentially on the modelling of the opponent's choices and decision patterns.

Opponent Modelling can be used as add-on to any approach to improve the quality of agent play. Opponent Modelling has been used to improve the play of Simulation based Poker agents, Equilibrium Strategies (making the agent diverge the equilibrium strategy) and Adaptive Imperfect Information Game-Tree Search [Bil06].

Another adaptive strategising that goes beyond opponent modelling is through past decisions evaluation. Statics tables or past cases can be adjusted to reflect the evaluation of a past decision so it can improve the successfulness of the decision mechanism.

### 3.1.5 Case-based Reasoning

Case-based reasoning, a methodology for solving problems by utilizing previous experiences [MDS00], has been used in a variety of games. The Case-based Reasoning methodology has been used successfully to play in Othello, Checkers, Real-Time Strategy computer games [SA09] [AMP05] and Poker [SR05] [ST06] [RW07].

Case-Based Reasoning is a methodology to use past solutions to solve new ones. It can be described as cycle (3.2) as new solutions are solved are introduced in the past solutions archive. Case-Based Reasoning can be described in a four step process [dMMB<sup>+</sup>05]:

- **Recall** — Given a current situation, or new case, a search is performed on the past cases solutions, or case base, for similar cases.
- **Reuse** — Analysing the past cases with the current situation a new solution to the current case can be created.
- **Revise** — The new solution is evaluated for its success.
- **Retain** — If the new solution is considered good and relevant (not very similar to present solutions) it is stored in the case base.

## State of Art

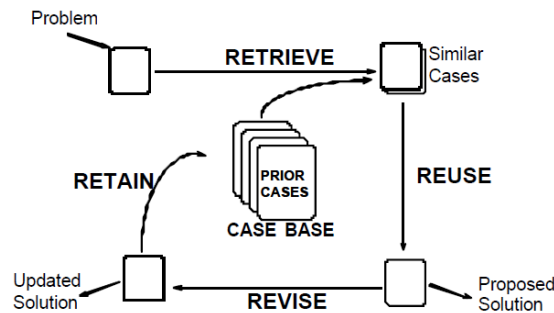


Figure 3.2: Case-Based Reasoning cycle. Taken from [dMMB<sup>+</sup>05].

The use of Case-Based Reasoning implies making two assumptions: similar cases have similar solutions and new solutions can be inferred from similar cases. These assumptions are also the crucial steps in CBR: the similarity comparison between cases and the search of similar cases, and also the adaptation of past solutions to produce a feasible new solution.

Three works using CBR were identified using poker as the main application. Two poker bots used a CBR methodology and a third work used CBR for opponent modelling. The CBR study for opponent modelling discussed the proposed approach of CBR against long-term statistical methods and came to conclusion that CBR was more unreliable [SR05]. It was discussed a modification in the CBR specification used to solve this problem but no future work encountered.

Three poker bots were identified that used Case-Based Reasoning at their core: Casey, Casper and Sartre. Both Casey and CASPER used similar methods in their implementation of CBR for Texas Hold'em Fixed Limit for full table's games. Sartre was design only for Heads-Up (two players) games.

Casey demonstrated that case-based Poker agent was capable of playing on par against four rule-based opponents but not a winner against more opponents [ST06]. Casey constructed its case base playing against RuleBot in Poker Academy in tables of 4, 6 and 8 players. Because it started with an empty case base it had a default playing method in it. This default mechanism, in this case, was to play randomly. Casey played evenly in the 4 player tables but lose in the 6 and 8 tables. Case-Based Reasoning did not yet proved to be able to play Poker at its fullest since full tables of nine or ten players are the most challenging ones.

CASPER [RW07], CASE-based Poker playER, which was developed after Casey used a slightly different approach. On contrary to Casey, CASPER, did not have strategies programmed into its case definition and did not started with an empty case-base. CASPER constructed its case base from PokiBot and SlimBot recorded games played in Poker Academy. Both PokiBot and SimBot have been proven to profit against human compe-

tion in the past [Dav02] and can be considered medium level players. CASPER was capable to play evenly in nine players table against the several bots in Poker Academy, namely PokiBot, SlimBot and JuggBot (tight rule based). It was argued that CASPER could perform well against humans and it was tested in play money games on the Internet against unknown human opponents and CASPER showed itself to be well profitable. CASPER was tested too with real money but it showed himself not to be competitive enough, losing money even if slowly.

Sartre [RW09b], Similarity Assessment Reasoning for Texas hold'em via Recall of Experience, is a recent work having the first related publication in 2009. Similar to Casey and Casper, Sartre plays the Fixed Limit variation of Texas Hold'em. Sartre case-base is generated by analysing the game logs of AAI Computer Poker Competition previous to 2009. It uses the game information that involves Hyperborean-eq bot which was the winner in 2008. Likely to Casper it was expected that Sartre would perform as well as Hyperborean-eq that document tests reveal that it performed worse. In 2009 Sartre already participated in the AAI Computer Poker Competition finishing in the 7th place of 13 participants [oA].

## 3.2 Online Poker

The most useful Poker room for Poker Bot research is the online world. The Internet provides a great intercommunication channel that allows a huge number of people to play poker against each other. The same channel of communication allows too that bots play against humans and against themselves too. An analyses of where poker it's played online is done in the next sections.

### 3.2.1 IRC games

The first mentioned online Poker games appeared in the IRC networks [Pap98]. Poker was played by user around the world with simpler IRC clients or with graphical interfaces that were developed [Mau]. It was always played for fun and not money but it different players even strong ones. There are still IRC poker games being played but the number of active players has been reduce to none relevant considering the players on online Poker sites. The message of currently active IRC network can be seen in figure 3.3.

### 3.2.2 Online Poker Sites

With the growing of Internet Poker users in late 90's soon someone realized the commercial potential in it. In January first of 1998 the first online poker money game was dealt [Wik]. Planet Poker marked an industry first, it was the first online Poker room with real money being played. From some years it showed a sustainable growth without

```

I have 5 clients and 1 servers
Current local users: 5 Max: 8
Current global users: 13 Max: 16
Highest connection count: 9 (8 clients) (50 connections received)
- irc.ircpoker.com Message of the Day -
-----
      A      A      A      A      A
     (V)    (V)    (V)    (V)    (V)
    (V)    (V)    (V)    (V)    (V)
   (V)    (V)    (V)    (V)    (V)
  (V)    (V)    (V)    (V)    (V)
 (V)    (V)    (V)    (V)    (V)
(V)    (V)    (V)    (V)    (V)
-----
                               hjw
-----
- Welcome to ircpoker.com -- The Home of IRC Poker
End of /MOTD command.

```

Figure 3.3: An active IRC Poker network greeting message

competition. Today the market of online poker sites is different and more competitive one.

According to [Par] the five Poker sites with bigger user market share are:

- **PokerStars** — 37%
- **Full Tilt Poker** — 16%
- **iPoker Network** — 9% (Not actually a site but a group of sites with different frontends )
- **PartyPoker** — 8%
- **Ongame Network** — 5% (A group of sites as well)

All this Poker sites as well others less popular share a common user base estimated in 14 to 23 million [WW07] players. Such large user base means equivalent huge revenue for poker sites estimated in the billions (American billions) of dollars.

All the main sites provide a gaming environment through specific software provide by them. In the case of gaming networks the same of very similar software is used. This software usually provide an interface with a room selection, the lobby, where the player can select between different types of poker variations, different amount of blinds between real money and play money and even between ring games or tournaments. A example of a poker room selection is given in figure 3.4.

### 3.2.3 PokerTH

PokerTH is a recent open source project created in 2006 for poker play. Even if it is not a commercial poker site software it can be consider so as the software and online playing feature are similar to common poker site software.

PokerTh software runs on Microsoft Windows, Mac OS X and GNU/Linux which is makes it probably the most inter-operative poker software. Poker TH provides a simple

## State of Art



Figure 3.4: Full Tilt Poker lobby

interface and functionalities to play Texas Hold'em No Limit in a tournament fashion but restricted to a single table of 10 players. Texas Hold'em can be played solo against simple bots or in network. It can be automatically connected to server where games are continually run between human players using PokerTH. According to official site [Pock] statistics the last 12 months had an average of 250 players online playing with PokerTH which makes it a interesting option too.

### 3.3 Conclusions

The continued study of poker agent building in the academia and outside of it has give birth to different approaches to computer poker reasoning. Common approaches usually rely on the simpler poker variations because of the huge strategy complexity in the most elaborated poker variations. Even so, poker bots are proving to be able to compete against medium level players in the most complex poker variations.

The use of case-based reasoning in academic works resulted into three successful bots. All these three works rely in the simpler form of betting that is Fixed Limit. One bot, Casper, was able to be a winning player with human opponents for play money in online play and the latest developed bot achieve a good result in the AAI Computer Poker Competition. Previous poker bots implementations using a case-based reasoning approach allows a good expectation on the research of a Texas Hold'em No Limit playing bot.

An analysis was performed on where the online poker players gather so the online play capacity can be developed. The focus of the online play capacity development chosen is the online poker sites (which includes PokerTH).

This chapter provides the main rationality on the implementation describe in the next chapter.

## **Chapter 4**

# **PBot: A Case-Based Reasoning Poker Bot**

PBot, an abbreviation of Poker Bot, is the implementation result of this thesis work and is described throughout this chapter.

PBot has two objectives: to play against human players online in order to challenge the huge amount of human players online and to use a case-based reasoning methodology in the reasoning process. The specific object of concern in the online play feature has been defined as the online poker sites software (including PokerTH). The case-based reasoning method principal ideas have also been explained previously.

A concern and objective took in this agent implementation was to limit the less possible future modifications and improvements as this agent seeks to provide the ground work for a later better playing agent.

In the following sections the developed online play capacity and the reasoning process based on CBR is described. First it is described the obtained solution architecture that groups the overall poker agent.

### **4.1 Solution Architecture**

The solution architecture, although it is described first, is also a consequence of the later described decisions. It is explained here to allow a more complete comprehension of the solution obtained.

The interaction with PokerTH and the generic online poker sites, to comply with the objective to play online against other poker players, is performed through the use of Open-Holdem. The reasons for this use are described in the section dedicated to online play.

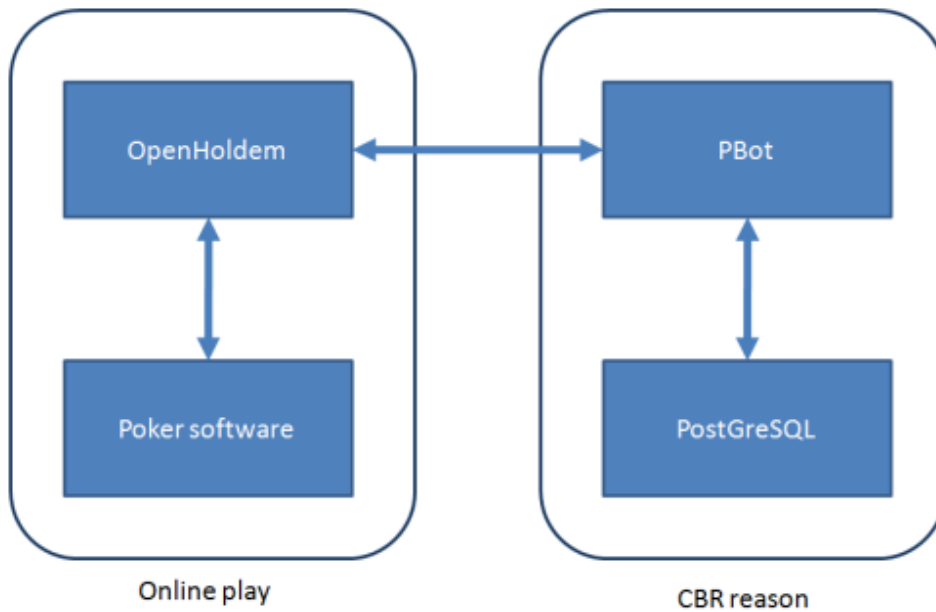


Figure 4.1: PBot architecture

PBot reason engine, following the method of case-based reasoning, is implemented through a TCP/IP server that provides poker gaming decisions in response to requests received with game situations or cases.

A loosely coupled integration between the online play capacity and the reason engine is chosen because it makes possible later use of different interaction mechanisms with players. This loosely coupled integration also permits that the development or modifications can be performed without the whole solution modification. This fact is especially relevant because OpenHoldem and PostgreSQL are provided by a third party. This type of integration allows a more independent development to third party restraints and an easier upgrade to newer versions.

This decision imposed the need to develop a protocol suitable for the purpose and is described later.

Figure 4.1 is exposes a schematic representation of the solution organization. OpenHoldem which interacts with poker sites software provides the online playing capacity to PBot. The poker agent reasoning using a case-based reasoning (CBR) methodology is performed by PBot while using PostgreSQL.

Portrayed the whole solution organization the online play feature and the PBot reason engine are described.



## 4.2 Online play

The online play feature is the capacity to interact with other players through third party software. These third party software have been decided to be poker sites software.

The software interaction with the poker sites software presents itself a problem as this type of software are solely design with an interaction concern with human counterparts. As such, this type of software doesn't make available any type of third party software interaction leaving only the option of emulate human input and perception.

The human input is received by the poker site software through keyboard and mouse data passed by the operating system. This information channel seems capable of being intercepted and manipulated because is directly manipulated by the operating system.

The operating system used in the development was Microsoft Windows and since most poker site software is only available for Windows no other operating system was even considered.

The Windows API provides a mechanism to programmatically control the global keyboard and mouse input so the question about emulating the user input can be easily solved through Windows API functions invocation. The remaining problem is how to express poker decisions through an emulated user input. In order to this remaining problem to be solved the human perception emulation problem has to be addressed.

### 4.2.1 Perception emulation

Poker site software is based on image representation of the game table, cards and other players' presence and actions. The input controls are contained in areas like buttons and text boxes. Because computer analysis is more demanding in resources, technology and advanced methods and also more propitious to errors it was not the first choice and other techniques were tested.

A common information acquisition technique from other process in Windows is through Windows API functions that provide functions that allow the retrieve of some information contained in Interface Controls provide by the Windows API. An Interface Control is a commonly used Interface object in Windows applications and some examples are text boxes, buttons, menus, scroll bars, dialog windows and other easily recognizable interface elements for a Windows user. Special interface controls can be developed but the common approach in Windows development is to use the controls provided by Windows. This permits exhibiting a recognizable interface for the user and saving developing resources.

A basic tool is provide by Microsoft allows to manually analyse the information that can be extracted through this technique is Microsoft Spy++. This program was tested with some poker site software and proved useless because the software did not expose that information, the software do not use the Windows provided interface controls.

## PBot: A Case-Based Reasoning Poker Bot



Figure 4.2: PokerTH interface, an example of standard poker site software interface

Another technique evaluated was Windows Inter-Process Communication (IPC) message interception, a common technique in debugging, which permits reliable information to be intercepted and processed. If the poker software utilized Windows IPC it would become possible to develop mechanisms to process that information and make it useful to PBot. This technique was test and proved not useful too.

There were reports, on some web sites, that Windows API hooking was successful to gather information out of some poker sites and it was tested too. Windows API hooking consists on intercepting Windows API function invocation by replacing the functions of the API with the desired functions. This method allows to collect data but also to manipulate it. In this case the data manipulation wasn't the objective but rather the data collection. In this case, the program WinAPIOverride32, which facilitates Windows API hooking, was used to evaluate the technique. There was some data of the poker software visible with Windows API hooking but no useful data was found.

Lastly image processing approach was taken. Because of the complexity of developing from zero an image processing engine some trials were done with OCR tools. This was experimented because all poker software utilized exhibits a textual description of the current game. This method could be prove useful if it could be used to get an accurate and on time text of the text log because it would allow to know the state of the game. OCR tools don't provide a complete accurate text extraction but the text to be processed was in plain computer type fonts so the accuracy was expected to be high. Even if the accuracy wasn't high enough several images with the same image could be processed so a more accurate text was available for processing.



Figure 4.3: Example of a log text box and below the text extracted by the OCR

Only two OCR tools were considered available for test, Tesseract OCR and a Microsoft Office Component Object Model (COM) that provided OCR functionalities. The Microsoft Office COM was generally considered of superior quality so it was chosen for test. The text extraction showed a good accuracy level (e.g. figure 4.3) but the technique exposed two problems. The first problem was the processing time. Both OCR tools only accepted images in a specific type so image conversion had to be employed and even a single image processing time was big enough to dismantle ideas of processing the same image more than once to improve the accuracy of the text obtained. The second problem detected was the possibility rapid updates in the text logs that could lead the information be lost resulting in a bot without accurate state information.

The only viable option available was to process image but in more efficient and simple methods. OpenHoldem offered just that, a more efficient, reliable and adaptable image processing tool for poker software. The use and explanation of the OpenHoldem is explained in the next section.

#### 4.2.2 OpenHoldem

OpenHoldem described in official web site "an open source screen scraping framework and programmable logic engine for the online Texas Holdem poker game" [Ope].

OpenHoldem does not implement a powerful enough programmable logic engine for a case-based reasoning bot but provides three useful features:

- An adaptable mechanism to percept poker software information (through image processing mainly)
- Functionality expansion through user crafted Dynamic-Link Libraries
- An engine to express poker decisions to poker site software (named Autoplayer)

## PBot: A Case-Based Reasoning Poker Bot



Figure 4.4: OpenScape main window with several fields defined

The above elicited functionalities provide the needed perception and input emulation and because of the functionality expansion allow a decision process to be transferred from the OpenHoldem to PBot decision engine.

The mechanism that implements the perception mechanism, usually named table scraping, does so following user defined rules in a file. This file, named table map, is specifically defined to each poker software interface and has to be updated on every interface modification. Although this limitation this parameter driven engine is powerful enough to read most of the current poker sites software. Besides allowing the table scrap of the poker software the table map also define the user input methods necessary for the user input emulation.

The table map is created manually with the OpenScape tool that is provide with OpenHoldem. With this tool the user can define areas of the poker table so the image acquisition can be easily processed and user emulated input can be generated.

The table map has two types of fields, information acquisition fields and input emulation fields. The input fields delimit the areas by its poker decision purpose and how that specific purpose can be expressed, through mouse click areas or keyboard input text fields. The information acquisition fields define what information to gather from a specific area of the image and how to process the image to retrieve the information.

The information acquisition (relative to image processing or not) can retrieve the following information from the poker site software:

- **Number of chairs** — The amount of chairs present in the table

- **Pot size** — Chip amount of the whole pot (chips that can be won in the current hand).
- **Community Cards** — The five community cards of the Flop, Turn and River.
- **Hand Number** — The number identification of current hand.
- **Last Hand Number** — The number identification of last hand played in the current table.
- **Betting rule** — The Texas Hold'em betting rule used in the current game table.
- **Small Blind Value** — The value of the small blind of the current game table.
- **Big Blind Value** — The big blind of the small blind of the current game table.
- **Players Names** — The name of each opponent player.
- **Players Pot** — The pot size of each opponent player.
- **Players exposed cards** — The cards revealed by other players (in the showdown or all-in).
- **Players seated** — Determine which game tables seats are occupied by opponent players.
- **Players in game** — Determine which players are currently in game (that did not fold).
- **Own player cards** — Identify which cards belong to the user.
- **Own player pot** — Identify the pot that corresponds to the user.
- **Action buttons** — Detecting the presence of Fold/Check/Call/Raise/All-in buttons and Swag (bet amount) controls.

The information that can be retrieved can be predefined in the table map (e.g. the number of table chairs), obtained through table window title parsing (e.g. the blinds values are usually present in the windows title) or through image processing. Most information retrieve relies on image processing.

The image processing fields can be of the following types: colour, text, image or colour hash. The colour fields perform a simple colour value comparison by comparing the current colour average value of the pixels in the area with a value present in the table map. This value comparison allow to extract information like the presence of seated player and if hasn't fold yet.

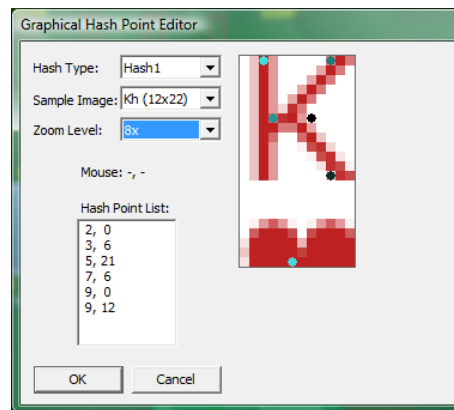


Figure 4.5: Colour hash graphical editor

The text field provide a simple OCR functionality and it's used to extract information only presented in text like the players names, their pot size and bet values. The text extraction is limited and only handles well monospaced fonts but it can extract the most relevant information like pot size. The text extraction relies on data comparison of the current image with the data of manually identified characters. To a complete text extraction all the characters possible to be exhibit (upper and lower case) have be manually captured and identified.

The image filed allows information extraction by image comparison. OpenHoldem compares the currently captured image to a set of images stored in the table map and returns the name of image. This image comparison can be used to identify the cards displayed by capturing all image cards and then by correctly place image fields.

Colour hash field resembles to the image field but instead of a complete image comparison a hash is used. The hash receives as parameters the colour values of selected pixels and compares it to a set of predefined values in the table map. The pixels used in hash are user predefined relative to the field area in the table map. An example of user defined pixels can be seen in the figure 4.5. This field, like the image field, is also used to determine the card exhibit. To this mean all cards images and later semi-automatic hash values have to be calculated.

As can be comprehensible from the description of the table map fields the creation of a single table map requires quite some work and in the event of a poker sites interface modification the table map has to be updated. Because the work required table map sharing is not common.

Two table maps were created, one for PokerTH and other second for a commercial poker site.

The PokerTH table map has 114 defined fields, the 52 cards images and their hash values and 148 characters data. The information captured by OpenHoldem using the created table map is represented in the figure 4.6.

## PBot: A Case-Based Reasoning Poker Bot

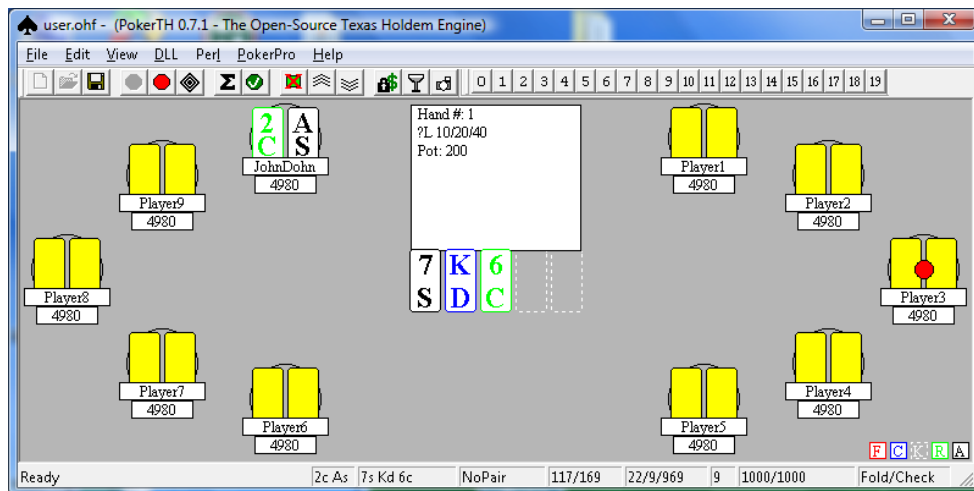


Figure 4.6: OpenHoldem capturing PokerTH table info

With the information capture and input methods defined the OpenHoldem library to provide a decision process transfer to PBot had to be developed. This library is named User DLL as is defined by OpenHoldem documentation.

### 4.2.3 User DLL

OpenHoldem programmable logic engine is suitable for rule based poker agents but not for a case-based reasoning one. Taking advantage of the functionality provide by OpenHoldem, specifically the information acquisition mechanism and the Autoplayer feature, requires too that the decision process is transferred to another software. In this case the software that will hold the decision process is the PBot server.

The OpenHoldem Autoplayer feature, which is the mechanism in charge of making the betting decisions, relies on OpenHoldem symbols values to decide which action to take. The OH Symbols which control the decision of OH AutoPlayer are `f$alli`, `f$swag`, `f$rais` and `f$call`. These symbols' values are determined by the OH script language. The Autoplayer executes the bot action according to the following rules which are evaluated from top down:

- **All-In** — If `f$alli` symbol evaluates to non zero and the all-in control is available an All-In is perform
- **Raise** — If `f$swag` is non zero and the swag control is available or if `f$rais` is non zero and the raise control is available a raise is perform
- **Call** — If `f$call` is non zero and the call control is available perform a call
- **Check** — If the check control is available perform a check

- **Fold** — If no other action was taken fold

In order to manipulate the OH AutoPlayer decisions the values of f\$alli, f\$swag, f\$rais and f\$call symbols have to be controlled by PBot. These symbols can be controlled from the User DLL if the OH script (the logic rules with defines the symbols values) loaded assign DLL defined symbols to those symbols.

The User DLL defines four symbols so that in order to control the Autoplayer feature. The OH Script only references the DLL symbols and does not have any other logic in it. The DLL symbols available to the OH Script are the strictly need, dll\$alli, dll\$swag, dll\$rais and dll\$call.

Although the OpenHoldem DLL interface is not very suitable to a case-based reasoning poker bot it was possible to adjust it to the purpose. The only DLL interface call, which masks varied functionalities, is the following:

```
process_message (const char* pmessage , const void* param)
```

This interface call masks several functionalities. A description of the relevant functionalities and arguments is given below.

- **state** — In each cycle of the table scrap OpenHoldem will send the current game stat to the DLL. The game state is provided in the param argument which will hold a C memory pointer to the structure representing that state (holdem\_state).
- **query** — The query message is the message sent to the User DLL when OpenHoldem tries to determine the symbols value with a "dll\$" prefix (e.g. dll\$alli). The User DLL should respond rapidly to such query as the OpenHoldem main thread is a waiting state.
- **pfgws** — pfgws message provides in the param argument the pointer to an internal OpenHoldem function. This function makes permits the User DLL to retrieve internal OpenHoldem symbols.

The data and interaction with the OpenHoldem is not very suitable for a PBot interaction because it lacks some event functionality and other information reporting. The User DLL development encountered two main problems, the data provided with the game data information was not enough and the dll\$ symbol invocation did not coincide with the PBot decision timings causing unnecessary calls.

The main game state information provided by OpenHoldem is provided with the structure holdem\_state. This structure is shown in the figure 4.7. The game state information does not provide information on what data was modified since the last call. As such the structure has to be analysed for changes to be able to react on those modifications so it can provide usable information to PBot.



## PBot: A Case-Based Reasoning Poker Bot

```
struct holdem_player {
    char          m_name[16]      ; //player name if known
    double        m_balance       ; //player balance
    double        m_currentbet    ; //player current bet
    unsigned char m_cards[2]     ; //player cards

    unsigned char m_name_known   : 1 ; //0=no 1=yes
    unsigned char m_balance_known : 1 ; //0=no 1=yes
    unsigned char m_fillerbits    : 6 ; //filler bits
    unsigned char m_fillerbyte    ; //filler bytes
};

struct holdem_state {
    char          m_title[64]     ; //table title
    double        m_pot[10]      ; //total in each pot

    unsigned char m_cards[5]     ; //common cards

    unsigned char m_is_playing   : 1 ; //0=sitting-out, 1=sitting-in
    unsigned char m_is_posting   : 1 ; //0=autopost-off, 1=autopost-on
    unsigned char m_fillerbits    : 6 ; //filler bits

    unsigned char m_fillerbyte    ; //filler byte
    unsigned char m_dealer_chair  ; //0-9

    holdem_player m_player[10]   ; //player records
};
```

Figure 4.7: struct holdem\_state

OpenHoldem, by its query message, query the dll\$ symbols in every scrap cycle and not only the Autoplayer need to play so the query message itself could not be considered a signalling for PBot decision and this presented another challenge.

The need to determine the correct decision moment is required by the case-based reasoning because it relies on case information, or in other words, on the game state, presented at the decision time. The correct decisions moment identification was achieved through the evaluation of the holdem\_state structure and the query of OH internal symbols.

The necessary game state information, or case information, is also obtain by holdem\_state data monitoring and internal OH symbols querying.

The User DLL was modelled to easy the development and maintenance of the code while obeying to its purpose to transfer to the PBot server the decision taking. The User DLL was then divided in three classes and the DLL interface handling.

The three classes developed are CClient, PBot and Bot. The Bot class intermediate the OH DLL interface handling and the PBot interface. It handles the information monitoring and rearranging to a PBot suitable form and controlling game states do correctly identify the poker decisions moments.

The PBot class provides the PBot reason engine functionalities. This class is responsible for argument marshalling and response unmarshalling and to detect communication errors. This is the class that implements the communication protocol of the PBot server.

## PBot: A Case-Based Reasoning Poker Bot

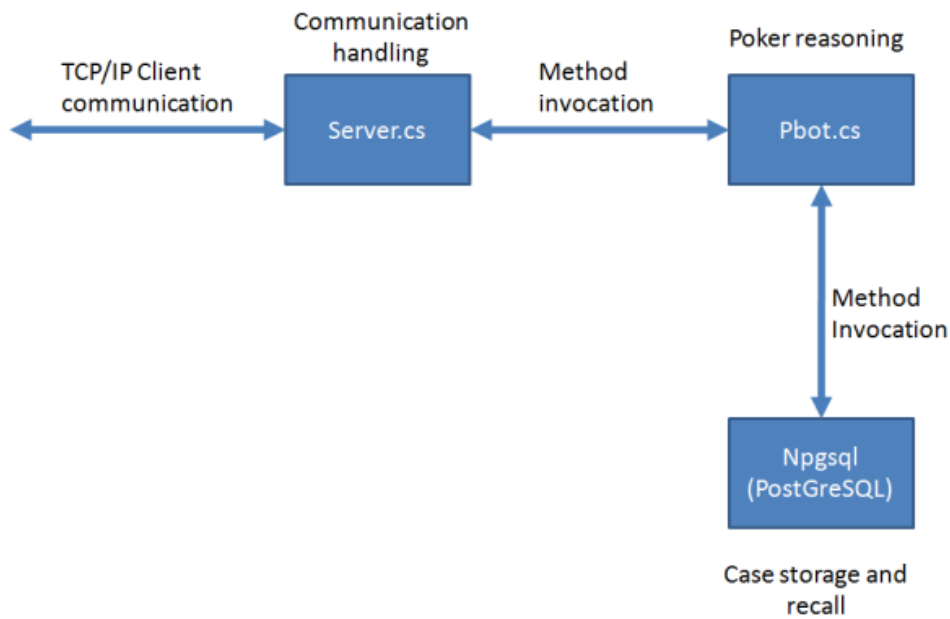


Figure 4.8: PBot core schematic organization

The CClient class is the class for establishing the communication TCP/IP channel used by PBot class.

### 4.3 PBot reason engine

PBot reason engine is available through a TCP/IP connection through typical server-client request-response communication architecture. The PBot TCP/IP server can work with simultaneous clients permitting several poker games being played with the reason logic and case base.

The PBot reason engine being implemented in a TCP/IP server allows that different clients can access the reason engine. This is useful as the reason engine is not dependent on the interaction mechanism with poker sites software and can be used to implement the reason engine to different poker software besides poker sites software.

PBot server and reason engine is built in C# language for .Net 2.0 Framework. The language of choice was C# because of the easy in development and debugging provide by the language and the .Net tools like Microsoft Visual Studio. The PBot reason engine is portrayed in the figure 4.8.

The server and the communication protocol are described in the next section. The reason engine is exposed after.

### 4.3.1 Communication

The PBot communication is handled by the Server class. The communication handling includes parsing and validation of the messages received, invocation of the proper PBot.cs methods and sending a message with the method result. The Server is capable of multiple clients handling by the instantiation of different PBot objects. The protocol used is of binary byte protocol developed from scratch.

The PBot protocol is a simple message protocol. It was developed with a more utilitarian goal to the development than with a goal of channel efficiency or other communication channel optimizations. The efficiency of the protocol was not a concern because it's expected that the client and the server run on the same machine or at least in the same LAN.

The communication is designed to be Client-Server and Request-Answer type of communication meaning that the server is passive and only reacts to issues sent by the client. The client is responsible for issuing requests correctly to ensure a proper response and only if client provides the server with the correct information the reason engine can provide the correct decision.

The PBot protocol is also developed to allow being easily extensible so it's a not a restrictive protocol in the information and functionally present at the current time. Each message is only restricted, even if easily modifiable, by its maximum size, being that value currently 2046 bytes. The maximum size of the raw message size is actually 4096 bytes but the message has to be escaped to prevent that reserved bytes are used in the message payload, the 2046 bytes is the safe maximum size. The three reserved bytes are the message start byte, 0xFF, the message end byte, 0xFE, and the escape byte, 0xFD.

The implemented protocol messages are enumerated below:

- **HELLO** — Handshake message for protocol implementation versions control.
- **SETTABLEINFO** — Allows various types of table information to be defined. (Not currently used in the reason process.) The available information setting messages are TABLENAME, OWNPLAYERNAME, PLAYERNAME, GAMETYPE, SMALLBLIND, BIGBLIND and RAKE.
- **NEGATIVEACK** — Negative acknowledgement. A negative response to a request. Sent in the case of a communication or method invocation error.
- **POSITIVEACK** — Positive acknowledgement. Informs the correct execution of a method.
- **DEBUGMSG** — Debug messaging. Allows the client to send any string to be logged in the server.
- **HANDACTION** — Message related to poker game events (message subtypes described below)

- **STARHAND** — Message go signal the start of a hand.
- **SETHANDPHASE** — Message to set the current hand phase (Pre-Flop, Flop, Turn or Rive).
- **ENDHAND** — Message go signal the end of a hand.
- **FLOPCARDS** — Message to send the flops cards information. (Not currently used in the reason process).
- **TURNCARD** — Message to send the turn card information. (Not currently used in the reason process.)
- **RIVERCARD** — Message to send the river card information. (Not currently used in the reason process.)
- **SETCF** — Message to send the common fields of the case information.
- **SETHANDRANK** — Message to send the current hand rank value.
- **HANDWON** — Signaling message of a won hand.
- **HANDLOST** — Signaling message of a lost hand.
- **GETBOTACTION** — Message that requests a bot decision based on the current game state information.
- **BOTACTION** — Message sent by the server informing the poker decision.

#### 4.3.2 PBot

The PBot is the class that keeps the information about the game state, provide methods for the bot to play Poker and the mechanisms related to case-based reasoning. PBot In order to deal with Poker information the Poker environment was modelled into the C# classes, structures and enumerations. Not everything Poker related was modelled but the required abstraction for PBot to be able to play Texas Hold'em No Limit games was achieved.

The PBot provides essentially the following methods to client:

- **SetPBDBConfig()** — Allows the definition of the database used, permitting for different databases to be used for each case.
- **SetTableName()** — Method to allow the storage of the information related to the table
- **SetTableGame()** — Method to define the type of game. Extension meant for later specification of other variations of Texas Hold'em like Pot Limit and Fixed Limit.

- **StartNewHand()** — Method to inform PBot that a new hand has started. Useful for hand phase control and memory structures clean.
- **SetHandPhase()** — Method to define the various hand phases of a hand. It's required that PBot knows the correct hand phase so it can correctly access the data structures related to that hand phase.
- **EndCurrentHand()** — Method to signal PBot the end of the hand phase. When this happens a case is usually stored.
- **HandWon()** — Method to inform PBot that a hand was won so it can evaluate its decision.
- **HandLost()** — Method to inform PBot that a hand was lost so it can evaluate its decision.
- **SetCF()** — SetCF stands for Set Common Fields. This method provides PBot the common information required to the case reasoning.
- **SetHandRank()** — Method to set the hand rank value of the current hand.
- **GetBotAction()** — Method to request PBot a decision on the current hand. This decision can be Fold, Check or Bet. If Bet is the choice the bet amount is also provided.

Initially other methods were used but with the transition the change to a TCP/IP architecture they became obsolete.

The crucial method is the `GetBotAction()` which performs at request the decision on the best action based on the current state information that hold by PBot and its case base. The information need is provide by `SetCF()` and `SetHandRank()` methods. Because this method information is relative to the hand phase the proper hand phase has to be set by the client too. Another case relevant method is `HandWon()` and `HandLost()` because this method allow a basic outcome evaluation of the PBot decision that is relevant for the case information.

In a case-based reasoning process the generic steps taken are retrieve, reuse, revise and retain. The retrieve and reuse steps are taken in `GetBotAction()` because it's the method that collects pasts cases the reuses that information to construct a solution to the new case. Revise is performed in `HandWon()` and `HandLost()` methods as this methods are the evaluators of the case reuse. The retain of the new case is performed when `EndCurrentHand()` is invoked.

In the next sections each CBR step will be explained in more detail but first the case information is explained.

Table 4.1: Case-Based Reasoning method association

CBR Step	Method
Retrieve Reuse	GetBotAction()
Revise	HandWon()/HandLost()
Retain	EndCurrentHand()

### 4.3.3 Case specification

In case-based reasoning the definition of the fields comprising the case is crucial. The fields of a case need to represent, with varying degree of accuracy, the situations that the reasoning is intend to solve. A central decision in the case specification is the choice of the fields that form the case index. The case index is the central point of the case specification because the index is the information that pre-defines the outcome of the case. The index isn't required to be completely accurate but allow a good degree of case outcome prediction because it will be used for case similarity comparison.

The PBot case information intends to reflect the information of a Texas Hold'em hand but this information it's not concentrate in the same case but divided in four different types. The four different types of cases are Pre-Flop, Flop, Turn and River and there isn't information sharing between them. This division is intended to reflect the hand phase the differences in play and to allow different information to be store for each phase. A case that would hold the complete information of a hand was possible but two reasons favoured the separation. First the case would hold much more information and the processing of each case would require much more resources. Secondly most of the cases would waste memory space because the number of Pre-Flop phases that a player participates is far greater than the number of River phases he plays.

The table 4.2 shows the fields information that the different cases holds. All the four types of cases share these fields being the only difference in HandRank calculation. This case representation is similar to the previous approaches with the main difference being in NeedToCall and Bet fields. This difference comes from the bet limit that the previous approaches were focused on being different from PBot, Casey and CASPER were developed for Fixed Limit rules.

The HandRank regarding the Pre-Flop gives a numeric classification to the hole cards strength, and the Flop, Turn and River, HandRank gives a classification to the hand strength (the five card combination between the private and common cards). The Pre-Flop HandRank value is calculated by a fixed classification of 169 distinct hand types possible, 13 paired hands, 78 suited hands and 78 unsuited hands. This classification of hands can be consult in the appendix B.

The fields NeedToCall and Bet are always relative because the amount involved. The fields are defined this way to allow the reuse of cases even between pots of different

Table 4.2: Case fields

<b>Field</b>	<b>Description</b>
SubCaseID	<i>ID that identifies a case</i>
NumPlayers	<i>Number of players that started the current hand phase (2-10)</i>
RelativePosition	<i>Relative position of PBot to the dealer position (0-1)</i>
PlayersInAction	<i>Number of players, that at PBot play turn, did not fold (2-10)</i>
PlayersToAct	<i>Number of players that still need to make the bet decision (0-10)</i>
NeedToCall	<i>Relative amount need to make a Call. (Pot amount/Call amount)</i>
HandRank	<i>Hand rank</i>
CInd	<i>Calculated index (0-1)</i>
Action	<i>Fold/Check/Call/Raise</i>
Bet	<i>Relative amount bet. (Pot amount/Call amount)</i>
Outcome	<i>Evaluation of outcome of the action taken (0-1)</i>

volume. This can be counter-productive since human players usually change of behaviour according to pot size but makes possible for case reuse in more situations and for the bot develop prevalent game style that if works with higher pots will probably work with lower pots. Defining the values relatively also reflects a common approach for the risk/reward assessment by human poker players. The human player normally weights the risk and the amount he can lose against the possibility and the amount he can win.

The value NeedToCall is pre-determined but the Bet value reflects the relative value that was bet by the Bot in the case.

As can be observed not all fields are of the same type. Some fields represent information known at the time PBot is request a decision and other fields are determined after. There is also CInd and SubCaseID which are auxiliary values. The field NeedToCall is a information known but the Bet field is a decision of PBot. Action is other value of PBot decision. On other hand Outcome is a field of new case evaluation.

The outcome value is meant to provide an evaluation of the decision taken (Action and Bet). This field can hold a negative evaluation of decisions that result in losses or a positive evaluation a winning decision. This feedback field allows that a bad decision to be avoid in following cases and a good decision to be repeated.

CInd is the case index field. CInd it's only a calculated value to ease case comparison and recall and its formula can be changed when ever required. The only requirement for a formula update is the actualization of all store cases CInd value to be according to the new

Table 4.3: A simplified example of case recall

SELECT CAST( abs("CInd"-0.225) as real) as "Delta", "RelativePosition", "Action", "NeedToCall", "Bet","Outcome","HandRank" FROM "CasePreFlop" WHERE abs("CInd"-0.225)<=0.25 ORDER BY "Delta" LIMIT 10						
Delta	RelativePosition	Action	NeedToCall	Bet	Outcome	HandRank
0.004093	0.333333	k	0	0	0	31
0.007249	0.8	c	0.166667	0.166667	0	4
0.04926	0.6	c	0.666667	0.666667	0	28
0.092554	0.666667	c	0.666667	0.666667	1	34
0.153698	1	c	0.285714	0.285714	0	29
0.171696	0.166667	c	0.111111	0.111111	0	80
0.180572	0.166667	c	0.428571	0.428571	0	82
0.216272	0.7	c	0.222222	0.222222	0	60
0.217801	0.333333	k	0	0	0	81

formula. The formula 4.1 represents the current CInd calculation. CInd takes the value of two records that are considered the basic predictors by Poker players of a player capacity of winning a hand, the cards he holds and the position in the hand. This mathematical assessment of that judgement was determined by an empiric assessment of importance of each factor to wining a hand.

$$CInd = 0.25 * RelativePosition + 0.75 * (HandRank/169) \quad (4.1)$$

#### 4.3.4 Case Recall

The case recall can be processing bottleneck operation because with it faces the possibility of searching between thousands of cases to encounter the similar cases to a current case. This fact was a contributing factor to DataBase Management System to be used and the CInd field to be present in the case information.

DBMS already are designed to deal with huge data providing at the same time tools to search and store information efficiently. DMS developed because of need to search records in the fastest way possible search algorithms and indexes to aid the search. The search optimizations of PostgreSQL and the addition of the CInd field offered a good opportunity for the case storage of thousands of cases without a severe setback in performance.

A Case recall is then executed by a SQL search. An example of an case recall is showed in table 4.3.

#### 4.3.5 Case Reuse

With the data of the cases recall the new case solution has to be obtained. In PBot the reasoning or case reuse is performed by determining a probability variable from the past



## PBot: A Case-Based Reasoning Poker Bot

```
float foldF = 0.0F, callA = 0.0F;
int callAC = 0;
foreach (DataRow dr in dt.AsEnumerable())
{
    if (dr.Field<string>("Action") != "f" && dr.Field<float>("Outcome") > 0.0F)
    {
        //if bet and won
        foldF += dr.Field<float>("Delta");
        callA += dr.Field<float>("Bet");
        callAC++;
    }
    else
    {
        //fold or bet and lost
        foldF += (1.0F - dr.Field<float>("Delta"));
    }
}
callA = (callA / callAC);
foldF = (foldF / dt.Rows.Count);
```

Figure 4.9: The processing code of the cases

cases and then randomly choosing an action with the probability variable. The probability variable and the random choice provide a tool of past case with a limited degree of unpredictability that is desired in a poker player. The case probability variable is foldF which stand for Fold Factor.

In each decision PBot has two options fold, leaving the hand action and not risk losses, or bet, commit a chip amount to the amount and risk losses and profits. Because Check, Call and All-In are just a Bet call with different bet amounts this actions are considered part of the Bet. PBot has then only two options, fold or bet. FoldF variable reflects this dichotomy by representing the fold probability based on the past cases.

FoldF value is calculated by iterating through each of the cases recalled. Each case has a Delta value associated that it's a numeric difference between the case and the present case. Considering that each case has an action record which represent the action taken and the outcome record that holds the success of the action a case is consider to reinforce the fold decision if it was a fold decision or if the decision was a bet and the outcome was negative. The fold probability is reinforced each time by 1-Delta and restrained by Delta value. The final foldF value is the average of all values.

The bet decision is considered the opposite of the fold decision, the bet probability is evolves inversely of the fold probability. The bet amount, if a bet is chosen, is the value of the average of successful bets taken.

The figure 4.9 illustrates the code that determines the value of foldF.

PBot case database starts empty and because of that in the beginning there will be no cases to reuse. After some hands played PBot may not find cases similar to the case of a present situation so it has to have a fallback mechanism. This fallback mechanism consists in accessing the hand rank and evaluating if it is in the 30% higher hands. If the hand is in those hands PBot does the minimum call and fold otherwise.

#### 4.3.6 Case Revise

When PBot takes a decision, due the nature of incomplete information of Poker and its random elements, it cannot determine if the decision is good. To PBot evaluate its decision it has to wait to determine if it should had taken a different decision or made a good choice. Due to the difficulty of evaluating a Poker decision in the online Poker context, the positive or negative evaluation, when a bet is made, is performed by observed if the bot has won or lost the hand. In the case that fold was taken it's just considered not a bad because it's ever more difficult to the evaluation. This is a simple method and not completely reliable because the fold decision could mean a loss in profit and a bet could have been higher increasing the profits. This evaluation is taken since reinforcing profit and restraining losses is considered useful.

Future work should reflect on case revise as a better case evaluation can provide a great impact on the poker play. This off course would also impact in the case reuse as the outcome factor would be take into account.

#### 4.3.7 Case store

When a hand is ended, being PBot informed by `EndCurrentHand()` method, PBot checks if the case information is filled. If the case information was filled in the correct time the case index is determined and the case information is store the correct table.

The PBot client has to ensure that the correct information was fed to PBot and that the invocation of the `EndCurrentHand()` to provoke the storage of the new cases. If this is not obeyed the case information can be lost of if incorrectly information is sent to PBot it can undermine future PBot decisions.

### 4.4 Conclusions

In the previous chapters an explanation of the main constrains, requirements and options to the poker agent developed was given. In this chapter the consequent work performed is describe.

One of main challenges of a poker agent capable of playing against human players online is the requirement of interaction with poker sites software. The resolution of this difficulty is exposed and the use of OpenHoldem is explained by its capacities. The utilization of OpenHoldem is not very suitable for a case-based reasoning poker agent and these limitations are overcome through the development of a Dynamic-Link Library.

PBot is separated in two parts, the communication and the reason engine. The communication through a TCP/IP server provides PBot a multiple client capability and permits a future expansion to other software interactions besides OpenHoldem.

## PBot: A Case-Based Reasoning Poker Bot

PBot reasoning engine, which relies on a case-based reasoning methodology, is presented and the relevant implementation choices exposed. Case recall and case retain developed is performed through the use a database system namely PostGreSQL for the potential efficiency possible by its search and index optimizations. Case reuse and case specification give a greater emphasis to the card ranks hold by the player and his position. The reason for this decision is the previously described impact on the hands outcome of the cards value and the player position in the game table.

The implementation work performed results in a poker agent capable of interacting with OpenHoldem and an agent that uses the case-based reasoning methodology in its decision process.



## Chapter 5

# Results and Tests

In this chapter the various objectives of PBot are tested and documented. The complete test of a Poker bot is difficult because of the chance elements in Poker and the opponent's great influence in his results. This fact demands that a huge number of hands to be played to constrain the influence of chance over the bot play quality analyses. Testing the Poker bot against different types of players is also required to take into consideration the results according to the opponents faced.

Firstly the OpenHoldem capacity to extract information and interact with Poker software is evaluated. Secondly the integration of OpenHoldem and PostgreSQL with PBot is taken into account. Finally the capacity of poker play by PBot is reviewed.

### 5.1 Table Scrap

For PBot to contest against Human players OpenHoldem, it's Input/Output mechanism has to be able to successfully interact with Poker software. Two table maps were developed and tested, one for PokerTH software and other for NoiQPoker Poker site software.

The successful capture of the PokerTH has been exemplified in the figure 4.6. The OpenHoldem table scrap of NoiQPoker is exhibited in the figure 5.1 showing the original image and information extracted visible in the OH interface.

The information that OpenHoldem can successfully read from these two poker tables is:

- **Pot size** — The amount in chips of the whole pot.
- **Community Cards** — All the five community cards are correctly identified.
- **Hand Number** — The number identification of current hand (only present in NoiQPoker).

## Results and Tests



Figure 5.1: OpenHoldem information capture along NoiQPoker table image

- **Last Hand Number** — The number identification of last hand (only present in NoiQPoker).
- **Betting rule** — The No Limit type of betting rules is identified (pre-define value in the table map).
- **Small Blind Value** — The small blind value is correctly obtained (not by image processing but through window title parsing).
- **Big Blind Value** — The big blind value is correctly obtained (not by image processing but through window title parsing).
- **Players Names** — The name of opponent players is perfectly obtained in PokerTH but with less accuracy in NoiQPoker.
- **Players Pot** — The amount of opponent players is correctly obtained.
- **Players seated** — Correctly identifies if a player is occupies a "chair".
- **Players in game** — Correctly identifies if a player is currently in game.

## Results and Tests

```
16:15:45 |SetCF()
16:15:45 |SetCF() 10 0.6 10 4 NaN (Not a Number)
16:15:45 |SetTableGame()
16:15:45 |SetHandRank()
16:15:45 |SetHandRank() 18 handPhase:PREFLOP
16:15:45 |GetBotAction()
16:15:45 |RecallCases()
16:15:45 |Reason()
16:15:45 |GetDefaultPHAction()
```

Figure 5.2: PBot log

- **Own player cards** — Detects correctly bot own cards.
- **Own player pot** — Detects correctly bot own pot size.
- **Action buttons** — Correctly detects the presence of Fold/Check/Call/Raise/All-in buttons and Swag controls.

## 5.2 OpenHoldem & PBot interaction

OpenHoldem communication with PBot is processed with a TCP/IP connection. TCP connection reliability doesn't make expectable that communication errors are encountered. The PBot server during development was tested with four client connections and no error was noticed. OpenHoldem, through the User.dll, successfully sent and received information related to Poker play. Due to OpenHoldem information gathering and the process that passes it to User.dll not being completely error prone some cases of unreliable information was detect in PBot records of operation. A log record with bad information is identified in the figure 5.2. In the case identified PBot fallback to a default action because it lacked the relative position information.

The error showed in figure 5.2 was not recurrent and OpenHoldem showed capable of table scrapping correctly. OH and the User.dll also were capable of executing PBot decisions. Only when for some reason AutoPlayer couldn't detect the correct control, the call button for example, it would fall back on folding.

## 5.3 PostgreSQL integration

Regarding the Postgres integration it was not possible to test the efficiency of the case recall solution in the presence of a huge volume of cases but it was successful test that it would retrieve the few cases present in less than one second (log time precision limited). The case storage was correctly managed too.

## 5.4 Poker play

PBot capacity of play was test in PokerTH and also briefly in NoiQPoker. PokerTH was the software privileged because on the same interface PokerTH also allows to play solo games against simple bots. This was preferred to human players because it made possible much faster tests. The first tests were intended to evaluate if PBot was capable of playing and consequent tests to provide simple considerations about how it plays.

Testing with PokerTH proved successful. PBot was capable of make decisions in its core and these being carried out by OpenHoldem. No problem was detected with the play of the bot in PokerTH against human and bots opponents. The tests with NoiQPoker were similar although few.

Although only simpler tests were taken it was detected a tendency in PBot play. With an empty case base, it always started to play taking few risks according its default decision mechanism. When the first's cases of folding, or bet losses, were recorded automatically PBot started to fold most of the hands. Even good hands were fold at this stage. It's argued that this was due to a bad selection process of similar cases.

## 5.5 Conclusions

This chapter exhibits the tests taken to evaluate this dissertation choices and the implementation of PBot.

PBot is considered able to interact with different poker software to play poker through OpenHoldem. OpenHoldem successfully reads the table information and is able to send poker play decisions to the poker software. The interaction between OpenHoldem and PBot, required for a successful implementation of this bot, is considered to be functional.

Case-based reasoning steps implementation could not be well evaluated as it was possible to test with a relevant number of played hands so simple tests and their results evaluation was perform. The case store and recall does not provide bad expectations in a complete test of these functionalities. Case reuse and revise although simple exposes a working implementation of case-based reasoning use.

The poker software interaction is considerable successful but a complete test on the reason process and the surrounding choices taken is still need even if the current implementation is a working example of the case-based reasoning methodology.



## Chapter 6

# Conclusions and Future Work

The goal of this work was to create a Poker agent that could play poker by itself and could evolve in time and through the chapters the process of developing this bot was described.

First to better comprehend the domain an analysis of poker and comprehensive description of poker was performed. The in depth description of poker is vital because of the variations of the game that exist today. All this variations have characteristics that change completely or totally the way poker is played. Texas Hold'em No Limit was taken as a challenge to extend the previous poker agents approach to the No Limit betting rules. Because is more challenging and provides a better quality feedback mechanism PBot was also meant to play against humans, which still are the better players in Poker Texas Hold'em No Limit.

All Artificial Intelligence Poker play approaches that were taken before present disadvantages and advantages. Some approaches, the ones considered most relevant, were evaluated briefly to identify this advantages, disadvantages and constrains within them. One approach in specific was taken in a deeper analyse because it gave the most promising achievements to the goals with the restrains of this endeavour. This approach was the Case-Base Reasoning.

With the main abstraction mechanism chosen the implementation design options and intrinsic specifications were described. The OpenHoldem use and its limitations were overcome and PBot was able to make betting decisions on the games it participated during trials. PBot was able to Recall, to Reason, Revise and to Retain the game states.

### 6.1 Goal achievements & Future Work

Being the main goal proposed by this dissertation the design and development of Poker bot able to play online and against human opponents, this goal can be said achieved. The

## Conclusions and Future Work

rationality over the case-based utility to Texas Hold'em No Limit was proven useful to PBot. There was however some long term goals that were not completely accomplished that can be achieved in future.

PBot, like any other of its precedents, cannot be considered a complete work but rather a step into the future. It lacks some functionality and has not yet proven to be a winner. The rationality of Case-Based Reasoning exposes that if PBot plays against good opposition it can learn from losing and winning against them to become a better player because of its memory use. PBot should then be tested increasingly with better opponents to test this idea and solve any problems encountered in the way.

The PBot project and development came to give strength to a sentence encountered when surfing the web about poker bots: "Nothing with this much earning potential is easy". The Poker world is a vast and it requires many hours to develop a good understanding of it. The goals were achieve but not enough development and testing was done to conclude PBot work. This dissertation should then be considered a step PBot full autonomous. Like it's said by some, Poker was meant to be better understood while playing and PBot is waiting to play against the best.

# References

- [AMP05] David W. Aha, Matthew Molineaux, and Marc Ponsen. Learning to win: Case-based plan selection in a real-time strategy game. *International Conference on Case-Based Reasoning*, pages 5–20, 2005.
- [Ant10] André Antunes. A febre do póquer, February 2010. Available online at <http://sic.sapo.pt/online/video/informacao/Reportagem+SIC/2010/2/a-febre-do-poquer09-02-2010-115921.htm>.
- [BBD<sup>+</sup>03] D. Billings, N. Burch, A. Davidson, R. Holte, J. Schaeffer, T. Schauenberg, and D. Szafron. Approximating game-theoretic optimal strategies for full-scale poker. *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 661–668, 2003.
- [Bil06] Darse Billings. *Algorithms and Assessment in Computer Poker*. PhD thesis, Department of Computing Science, University of Alberta, Canada, 2006.
- [BPSS99] Darse Billings, Lourdes Peña, Jonathan Schaeffer, and Duane Szafron. Using probabilistic knowledge and simulation to play poker. *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 697–703, 1999.
- [Dav02] Aaron Davidson. Opponent modeling in poker: Learning and acting in a hostile and uncertain environment. Master’s thesis, Department of Computing Science, University of Alberta, 2002.
- [dC09] Nuno Pedro Silva da Cruz. Using a high-level language to build a poker playing agent. Master’s thesis, Faculdade de Engenharia da Universidade do Porto, 2009.
- [dCM08] Pedro Daniel da Cunha Mendes. High-level language to build poker agents. Master’s thesis, Faculdade de Engenharia da Universidade do Porto, 2008.
- [Dev] James Devlin. How i built a working poker bot, part 1. <http://www.codingthewheel.com/archives/how-i-built-a-working-poker-bot>, Consulted on February, 2010.
- [dMMB<sup>+</sup>05] Ramon López de Mántaras, David McSherry, Derek Bridge, David Leake, Barry Smyth, Susan Craw, Boi Faltings, Mary Lou Maher, Michael Cox, Kenneth Forbus, Mark Keane, Agnar Aamodt, , and Ian Watson. Retrieval,

## REFERENCES

- reuse, revision, and retention in case-based reasoning. *The Knowledge Engineering Review*, 20(3):215–240, 2005.
- [Get] Getpokernews.com. Online poker revenue statistics. <http://getpokernews.com/poker-news/online-poker-revenue-statistics/>, Consulted on February, 2010.
- [Mau] Michael Maurer. What is irc poker and how can i play? <http://www.conjelco.com/faq/ircpoker.html>, Consulted on February, 2010.
- [Max] MaxinMontreal. Maxinmontreal forums. <http://www.maxinmontreal.com/forums/>, Consulted on February, 2010.
- [MDS00] Julie Main, Tharam Dillon, and Simon Shiu. A tutorial on case-based reasoning. *Soft computing in case based reasoning*, pages 1–28, 2000.
- [oA] University of Alberta. Slideshow of the results. <http://webdocs.cs.ualberta.ca/~pokert/2009/results/presentation.htm>, Consulted on February, 2010.
- [Ope] OpenHoldem. Openholdem official web site. <http://code.google.com/p/openholdembot/>, Consulted on February, 2010.
- [Pap98] Denis Richard Papp. Dealing with imperfect information in poker. Master’s thesis, Department of Computing Science, University of Alberta, 1998.
- [Par] PartyGaming. Partygaming financial report 2009. [http://www.partygaming.com/prty/en/investors/financialperformance/fp\\_financialreports/](http://www.partygaming.com/prty/en/investors/financialperformance/fp_financialreports/), Consulted on February, 2010.
- [Pen99] Lourdes Pena. Probabilities and simulations in poker. Master’s thesis, Department of Computing Science, University of Alberta, 1999.
- [Poka] PokerAI. Pokeraï forums. <http://pokeraï.org/pf3/>, Consulted on February, 2010.
- [Pokb] PokerListings. About chris moneymaker. [http://www.pokerlistings.com/poker-player\\_chris-moneymaker](http://www.pokerlistings.com/poker-player_chris-moneymaker), Consulted on February, 2010.
- [Pokc] PokerTH. Pokerth. <http://www.pokerth.net/>, Consulted on February, 2010.
- [Que] El Quebrado. Pokerversions.png. <http://commons.wikimedia.org/wiki/File:Pokerversions.png>, Consulted on February, 2010.
- [RW07] Jonathan Rubin and Ian Watson. Investigating the effectiveness of applying case-based reasoning to the game of texas hold’em. *Proceedings of the 20th Florida Artificial Intelligence Research Society Conference*, pages 417–422, 2007.

## REFERENCES

- [RW09] Jonathan Rubin and Ian Watson. A memory-based approach to two-player texas hold'em. *AI '09: Proceedings of the 22nd Australasian Joint Conference on Advances in Artificial Intelligence*, pages 465–474, 2009.
- [SA09] Tomasz Szczepański and Agnar Aamodt. Case-based reasoning for improved micromanagement in real-time strategy games. *Workshop on Case-Based Reasoning for Computer Games, of the Eighth International Conference on Case-Based Reasoning*, 2009. Available online at <http://gaia.fdi.ucm.es/cbrcg09/>.
- [Sk104] David Sklansky. *The Theory of Poker*. Two Plus Two Publishing, sixth edition, 2004.
- [SR05] M. Salim and P. Rohwer. Poker opponent modeling, 2005. Available online at <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.90.578>.
- [ST06] Arild Sandven and Bjørnar Tessem. A case-based learner for poker. *The Ninth Scandinavian Conference on Artificial Intelligence*, pages 159–167, 2006.
- [Wik] Wikipedia. Planet poker. [http://en.wikipedia.org/wiki/Planet\\_Poker](http://en.wikipedia.org/wiki/Planet_Poker), Consulted on February, 2010.
- [WW07] Robert T. Wood and Robert J. Williams. Internet gambling: Past, present and future. *Research and measurement issues in gambling studies*, pages 491–514, 2007.

## REFERENCES

## Appendix A

# Glossary of Poker Terms

This appendix provides a description of the Poker terms used in this dissertation and some others. Glossary adjusted from [dCM08].

- **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.
- **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 \$10-\$20 Hold'em, 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.
- **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) (vt) 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 the intention of raising in the same betting round after an opponent bets.
- **Community Cards** — The public cards shared by all players.

## Glossary of Poker Terms

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



## Glossary of Poker Terms

- **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).
- **Pre-flop** — 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).
- **Raise** — To increase the current level of betting. If the current level of betting is zero, the term bet is preferred.
- **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
- **Re-raise** — To increase to the third level of betting after a bet and a raise.
- **River** — The fifth community card dealt in Hold'em, followed by the fourth (and final) betting round.
- **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. A very powerful and well-disguised hand (compare trips).
- **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.
- **Small Bet** — The smallest bet size in Limit poker (e.g., \$10 in \$10-\$20 Hold'em).
- **Small Blind** — A forced bet made before the deal of the cards (e.g., \$5 in \$10-\$20 Hold'em, 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).
- **Suited** — Two cards of the same suit, such as both Hearts. More likely to make a flush than other combinations (compare offsuit or unsuited).
- **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).

## Glossary of Poker Terms

- **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.
- **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 a lot of raising and re-raising. Also called an action game.

## Appendix B

# 169 Hole Cards Rank

Hole cards rank, taken from OpenHoldem Manual.

Number of opponents on the X axis, hand rank (1 is best) on the Y axis

	9	8	7	6	5	
1	*AAo*- 12	*AAo*- 12	*AAo*- 12	*AAo*- 12	*AAo*- 12	1
2	*KKo*- 24	*KKo*- 24	*KKo*- 24	*KKo*- 24	*KKo*- 24	2
3	*QOo*- 36	*QOo*- 36	*QOo*- 36	*QOo*- 36	*QOo*- 36	3
4	*AKs*- 44	*AKs*- 44	*JJo*- 48	*JJo*- 48	*JJo*- 48	4
5	*JJo*- 56	*JJo*- 56	*AKs*- 56	*AKs*- 56	*AKs*- 56	5
6	*AQs*- 64	*AQs*- 64	*AQs*- 64	*AQs*- 64	*TTo*- 68	6
7	*KQs*- 72	*KQs*- 72	*KQs*- 72	*TTo*- 76	*AQs*- 76	7
8	*AJs*- 80	*AJs*- 80	*TTo*- 84	*KQs*- 84	*KQs*- 84	8
9	*TTo*- 92	*TTo*- 92	*AJs*- 92	*AJs*- 92	*AKo*- 108	9
10	*KJs*- 100	*AKo*- 116	*AKo*- 116	*AKo*- 116	*AJs*- 116	10
11	*AKo*- 124	*KJs*- 124	*KJs*- 124	*KJs*- 124	*99o*- 128	11
12	*ATs*- 132	*ATs*- 132	*ATs*- 132	*ATs*- 132	*KJs*- 136	12
13	*QJs*- 140	*QJs*- 140	*QJs*- 140	*QJs*- 140	*ATs*- 144	13
14	*KTs*- 148	*KTs*- 148	*KTs*- 148	*99o*- 152	*QJs*- 152	14
15	*QTs*- 156	*QTs*- 156	*99o*- 160	*KTs*- 160	*AQo*- 176	15
16	*JTs*- 164	*99o*- 168	*QTs*- 168	*AQo*- 184	*KTs*- 184	16
17	*99o*- 176	*JTs*- 176	*AQo*- 192	*QTs*- 192	*KQo*- 208	17
18	*AQo*- 200	*AQo*- 200	*JTs*- 200	*JTs*- 200	*QTs*- 216	18
19	*A9s*- 208	*KQo*- 224	*KQo*- 224	*KQo*- 224	*88o*- 228	19
20	*KQo*- 232	*A9s*- 232	*A9s*- 232	*88o*- 236	*JTs*- 236	20
21	*88o*- 244	*88o*- 244	*88o*- 244	*A9s*- 244	*AJo*- 260	21
22	*K9s*- 252	*K9s*- 252	*AJo*- 268	*AJo*- 268	*A9s*- 268	22
23	*T9s*- 260	*T9s*- 260	*K9s*- 276	*K9s*- 276	*KJo*- 292	23
24	A8s*- 268	*AJo*- 284	*A8s*- 284	*KJo*- 300	*K9s*- 300	24
25	J9s*- 276	*A8s*- 292	*T9s*- 292	*A8s*- 308	*A8s*- 308	25
26	Q9s*- 284	J9s*- 300	*KJo*- 316	*Q9s*- 316	*QJo*- 332	26
27	77o*- 296	Q9s*- 308	*Q9s*- 324	*QJo*- 340	*ATo*- 356	27
28	AJo*- 320	KJo*- 332	J9s*- 332	*T9s*- 348	*77o*- 368	28
29	A5s*- 328	77o*- 344	QJo*- 356	*J9s*- 356	*Q9s*- 376	29
30	A7s*- 336	A5s*- 352	77o*- 368	ATo*- 380	*T9s*- 384	30
31	A4s*- 344	A7s*- 360	A7s*- 376	77o*- 392	*J9s*- 392	31
32	KJo*- 368	QJo*- 384	A5s*- 384	A7s*- 400	*KTo*- 416	32
33	A3s*- 376	A4s*- 392	ATo*- 408	KTo*- 424	*A7s*- 424	33
34	66o*- 388	A3s*- 400	A4s*- 416	A5s*- 432	*A5s*- 432	34
35	A6s*- 396	A6s*- 408	KTo*- 440	A4s*- 440	QTo*- 456	35
36	QJo*- 420	ATo*- 432	A6s*- 448	A6s*- 464	JTo*- 480	36
37	K8s*- 428	K8s*- 440	K8s*- 456	K8s*- 472	K8s*- 488	37
38	A2s*- 436	66o*- 452	A3s*- 464	A6s*- 480	A4s*- 496	38
39	T8s*- 444	A2s*- 460	JTo*- 488	JTo*- 504	A6s*- 504	39

## 169 Hole Cards Rank

40	98s - 452	T8s - 468	QTo - 512	A3s - 512	66o - 516	40
41	J8s - 460	KTo - 492	66o - 524	66o - 524	A3s - 524	41
42	ATo - 484	98s - 500	T8s - 532	T8s - 532	Q8s - 532	42
43	Q8s - 492	Q8s - 508	A2s - 540	Q8s - 540	T8s - 540	43
44	55o - 504	J8s - 516	Q8s - 548	A2s - 548	K7s - 548	44
45	K7s - 512	JTo - 540	98s - 556	J8s - 556	J8s - 556	45
46	JTo - 536	QTo - 564	J8s - 564	K7s - 564	98s - 564	46
47	KTo - 560	K7s - 572	K7s - 572	98s - 572	A2s - 572	47
48	44o - 572	55o - 584	K6s - 580	K6s - 580	A9o - 596	48
49	33o - 584	87s - 592	55o - 592	55o - 592	K6s - 604	49
50	22o - 596	K6s - 600	87s - 600	A9o - 616	K9o - 628	50
51	QTo - 620	44o - 612	K5s - 608	87s - 624	55o - 640	51
52	87s - 628	97s - 620	97s - 616	K5s - 632	K5s - 648	52
53	K6s - 636	33o - 632	44o - 628	97s - 640	87s - 656	53
54	97s - 644	22o - 644	K4s - 636	Q7s - 648	Q7s - 664	54
55	K5s - 652	K5s - 652	T7s - 644	K4s - 656	A8o - 688	55
56	76s - 660	T7s - 660	Q7s - 652	T7s - 664	Q9o - 712	56
57	T7s - 668	K4s - 668	J7s - 660	K9o - 688	T7s - 720	57
58	K4s - 676	76s - 676	K3s - 668	J7s - 696	97s - 728	58
59	K3s - 684	Q7s - 684	A9o - 692	K3s - 704	K4s - 736	59
60	Q7s - 692	K3s - 692	76s - 700	T9o - 728	T9o - 760	60
61	K2s - 700	J7s - 700	33o - 712	44o - 740	J7s - 768	61
62	J7s - 708	K2s - 708	K2s - 720	Q6s - 748	J9o - 792	62
63	86s - 716	86s - 716	Q6s - 728	Q9o - 772	K3s - 800	63
64	65s - 724	65s - 724	22o - 740	J9o - 796	Q6s - 808	64
65	54s - 732	Q6s - 732	K9o - 764	K2s - 804	44o - 820	65
66	Q6s - 740	54s - 740	86s - 772	76s - 812	K2s - 828	66
67	75s - 748	A9o - 764	T9o - 796	A8o - 836	76s - 836	67
68	Q5s - 756	Q5s - 772	65s - 804	86s - 844	Q5s - 844	68
69	96s - 764	T9o - 796	Q5s - 812	Q5s - 852	A7o - 868	69
70	Q4s - 772	96s - 804	J9o - 836	33o - 864	86s - 876	70
71	Q3s - 780	75s - 812	Q9o - 860	65s - 872	A5o - 900	71
72	64s - 788	Q4s - 820	96s - 868	Q4s - 880	Q4s - 908	72
73	T9o - 812	K9o - 844	54s - 876	96s - 888	96s - 916	73
74	Q2s - 820	J9o - 868	Q4s - 884	22o - 900	T6s - 924	74
75	A9o - 844	Q3s - 876	A8o - 908	T6s - 908	33o - 936	75
76	T6s - 852	64s - 884	75s - 916	Q3s - 916	K8o - 960	76
77	53s - 860	T6s - 892	T6s - 924	A7o - 940	J6s - 968	77
78	J6s - 868	Q9o - 916	Q3s - 932	J6s - 948	65s - 976	78
79	85s - 876	Q2s - 924	J6s - 940	75s - 956	Q3s - 984	79
80	J9o - 900	J6s - 932	Q2s - 948	54s - 964	A6o - 1008	80
81	K9o - 924	85s - 940	64s - 956	Q2s - 972	A4o - 1032	81
82	43s - 932	53s - 948	85s - 964	A5o - 996	T8o - 1056	82
83	J5s - 940	A8o - 972	J5s - 972	J5s - 1004	J5s - 1064	83
84	Q9o - 964	J5s - 980	53s - 980	K8o - 1028	Q8o - 1088	84
85	74s - 972	J4s - 988	A7o - 1004	85s - 1036	Q2s - 1096	85
86	J4s - 980	74s - 996	J4s - 1012	64s - 1044	75s - 1104	86
87	J3s - 988	43s - 1004	A5o - 1036	A4o - 1068	J8o - 1128	87
88	J2s - 996	J3s - 1012	J3s - 1044	T8o - 1092	98o - 1152	88
89	95s - 1004	95s - 1020	95s - 1052	J4s - 1100	54s - 1160	89
90	63s - 1012	J2s - 1028	74s - 1060	A6o - 1124	22o - 1172	90
91	A8o - 1036	63s - 1036	T8o - 1084	98o - 1148	A3o - 1196	91
92	T5s - 1044	T5s - 1044	J2s - 1092	Q8o - 1172	K7o - 1220	92
93	52s - 1052	A5o - 1068	43s - 1100	J8o - 1196	J4s - 1228	93
94	42s - 1060	A7o - 1092	A4o - 1124	J3s - 1204	85s - 1236	94
95	T4s - 1068	T8o - 1116	K8o - 1148	53s - 1212	J3s - 1244	95
96	T3s - 1076	T4s - 1124	T5s - 1156	A3o - 1236	64s - 1252	96
97	84s - 1084	98o - 1148	98o - 1180	95s - 1244	95s - 1260	97
98	98o - 1108	T3s - 1156	J8o - 1204	J2s - 1252	A2o - 1284	98
99	T2s - 1116	84s - 1164	T4s - 1212	T5s - 1260	T5s - 1292	99

## 169 Hole Cards Rank

100	A5o -1140	52s -1172	63s -1220	74s -1268	J2s -1300	100
101	T8o -1164	A4o -1196	A6o -1244	K7o -1292	K6o -1324	101
102	A7o -1188	T2s -1204	A3o -1268	T4s -1300	T4s -1332	102
103	73s -1196	42s -1212	Q8o -1292	43s -1308	53s -1340	103
104	32s -1204	K8o -1236	T3s -1300	A2o -1332	74s -1348	104
105	A4o -1228	A3o -1260	84s -1308	T3s -1340	87o -1372	105
106	94s -1236	J8o -1284	T2s -1316	84s -1348	T3s -1380	106
107	93s -1244	A6o -1308	52s -1324	63s -1356	97o -1404	107
108	62s -1252	73s -1316	A2o -1348	T2s -1364	43s -1412	108
109	A3o -1276	Q8o -1340	K7o -1372	87o -1388	T7o -1436	109
110	K8o -1300	94s -1348	42s -1380	K6o -1412	T2s -1444	110
111	J8o -1324	32s -1356	87o -1404	52s -1420	84s -1452	111
112	92s -1332	93s -1364	94s -1412	94s -1428	K5o -1476	112
113	A6o -1356	A2o -1388	73s -1420	97o -1452	Q7o -1500	113
114	87o -1380	87o -1412	93s -1428	93s -1460	63s -1508	114
115	Q8o -1404	92s -1420	92s -1436	73s -1468	J7o -1532	115
116	83s -1412	62s -1428	32s -1444	42s -1476	94s -1540	116
117	A2o -1436	K7o -1452	97o -1468	T7o -1500	K4o -1564	117
118	82s -1444	83s -1460	K6o -1492	K5o -1524	93s -1572	118
119	97o -1468	97o -1484	62s -1500	92s -1532	52s -1580	119
120	72s -1476	82s -1492	83s -1508	Q7o -1556	73s -1588	120
121	K7o -1500	76o -1516	T7o -1532	J7o -1580	Q6o -1612	121
122	76o -1524	72s -1524	82s -1540	32s -1588	76o -1636	122
123	T7o -1548	K6o -1548	76o -1564	76o -1612	92s -1644	123
124	65o -1572	T7o -1572	K5o -1588	83s -1620	K3o -1668	124
125	K6o -1596	65o -1596	J7o -1612	62s -1628	42s -1676	125
126	86o -1620	K5o -1620	72s -1620	K4o -1652	86o -1700	126
127	54o -1644	86o -1644	Q7o -1644	82s -1660	83s -1708	127
128	K5o -1668	J7o -1668	K4o -1668	86o -1684	K2o -1732	128
129	J7o -1692	54o -1692	86o -1692	Q6o -1708	Q5o -1756	129
130	Q7o -1716	Q7o -1716	65o -1716	K3o -1732	82s -1764	130
131	75o -1740	K4o -1740	K3o -1740	72s -1740	62s -1772	131
132	K4o -1764	K3o -1764	Q6o -1764	K2o -1764	32s -1780	132
133	K3o -1788	75o -1788	K2o -1788	65o -1788	96o -1804	133
134	96o -1812	K2o -1812	54o -1812	96o -1812	65o -1828	134
135	64o -1836	Q6o -1836	96o -1836	Q5o -1836	T6o -1852	135
136	K2o -1860	96o -1860	75o -1860	54o -1860	Q4o -1876	136
137	53o -1884	64o -1884	Q5o -1884	T6o -1884	J6o -1900	137
138	Q6o -1908	Q5o -1908	T6o -1908	75o -1908	72s -1908	138
139	85o -1932	T6o -1932	Q4o -1932	Q4o -1932	75o -1932	139
140	T6o -1956	53o -1956	64o -1956	J6o -1956	Q3o -1956	140
141	Q5o -1980	Q4o -1980	J6o -1980	Q3o -1980	54o -1980	141
142	43o -2004	85o -2004	Q3o -2004	Q2o -2004	J5o -2004	142
143	Q4o -2028	J6o -2028	85o -2028	85o -2028	Q2o -2028	143
144	Q3o -2052	Q3o -2052	53o -2052	64o -2052	85o -2052	144
145	74o -2076	Q2o -2076	Q2o -2076	J5o -2076	J4o -2076	145
146	Q2o -2100	43o -2100	J5o -2100	53o -2100	64o -2100	146
147	J6o -2124	74o -2124	74o -2124	J4o -2124	95o -2124	147
148	63o -2148	J5o -2148	43o -2148	95o -2148	J3o -2148	148
149	J5o -2172	J4o -2172	J4o -2172	J3o -2172	T5o -2172	149
150	95o -2196	95o -2196	95o -2196	74o -2196	53o -2196	150
151	52o -2220	63o -2220	J3o -2220	T5o -2220	J2o -2220	151
152	J4o -2244	J3o -2244	J2o -2244	43o -2244	74o -2244	152
153	42o -2268	J2o -2268	63o -2268	J2o -2268	T4o -2268	153
154	J3o -2292	52o -2292	T5o -2292	T4o -2292	43o -2292	154
155	J2o -2316	T5o -2316	52o -2316	63o -2316	T3o -2316	155
156	84o -2340	84o -2340	T4o -2340	T3o -2340	84o -2340	156
157	T5o -2364	42o -2364	84o -2364	84o -2364	63o -2364	157
158	32o -2388	T4o -2388	T3o -2388	52o -2388	T2o -2388	158
159	T4o -2412	T3o -2412	42o -2412	T2o -2412	94o -2412	159

### 169 Hole Cards Rank

160	T3o -2436	32o -2436	T2o -2436	94o -2436	52o -2436	160
161	73o -2460	T2o -2460	73o -2460	42o -2460	93o -2460	161
162	T2o -2484	73o -2484	94o -2484	73o -2484	73o -2484	162
163	62o -2508	94o -2508	32o -2508	93o -2508	42o -2508	163
164	94o -2532	62o -2532	93o -2532	92o -2532	92o -2532	164
165	93o -2556	93o -2556	62o -2556	32o -2556	83o -2556	165
166	92o -2580	92o -2580	92o -2580	83o -2580	62o -2580	166
167	83o -2604	83o -2604	83o -2604	62o -2604	32o -2604	167
168	82o -2628	82o -2628	82o -2628	82o -2628	82o -2628	168
169	72o -2652	72o -2652	72o -2652	72o -2652	72o -2652	169
	9	8	7	6	5	