

---

**BMI paper**

# **Application of AI in poker**

Author: Erwin Schuijtvlot  
Supervisor: Dr. Wojtek Kowalczyk

VU University Amsterdam  
Faculty of Sciences  
Study Business Mathematics and Informatics

De Boelelaan 1081a  
1081 HV Amsterdam

June 2011

---

## PREFACE

This paper was written as part of the Master Business Mathematics and Informatics study from the VU. The goal of writing this paper is to do an independent study of a subject where at least two of three domains mathematics, informatics and business are represented.

The subject of this paper is an application of AI techniques to poker. The main focus has been put on the informatics aspect with the secondary focus on the business aspect.

The subject was chosen because I am a recreational poker player myself and I had already done some small experiments with automated play.

For me the most enjoyable part of doing this research was performing the practical experiments and this is something I want to continue with for myself in the future.

I would like to thank Dr. Wojtek Kowalczyk for the good advice and support.

E. Schuijtvlot

June 2011

## SUMMARY

Poker has some properties which makes it an interesting game to study within AI. There is uncertainty, incomplete information, deception, a clear objective, one or more competing agents and a set of well defined rules.

There exists a diverse market for tools aimed at improving online poker play. This software is geared towards the 1.8 million poker players playing daily which generate an income of \$2.4 billion a year for the online casinos. Among this software there are only a couple of quality poker botting tools available.

A selection of algorithms that have been applied successfully to poker in the past are described. The most simple and straightforward of these algorithms are knowledge based. These algorithms directly use domain knowledge in the form of 'if-then' statements or formulas.

The second type of algorithms described is based on simulation where either off-line or on-line expected values of the available actions are estimated by means of Monte Carlo simulation.

The third type of algorithms are the game theoretical algorithms. These algorithms attempt to compute a so called near equilibrium strategy which is a strategy that cannot be exploited by other players and results possibly in winning. The main technique for finding these strategies is by solving a linear programming problem.

Finally an application of case based reasoning to poker is described, where an automated agent makes its decisions based on a database with game history.

The goal of the experiments done for this paper was to see whether a simple knowledge based agent was capable of competing against the weak opposition found in low stakes online poker tournaments. For these experiments the open source poker botting framework Open Holdem was used.

Two experiments were run, the first agent was based on a formula based system found in the poker literature. This agent was stopped after 50 games because it displayed unwanted behavior and incurred a small loss. A second attempt was based on simple tables found in a book specifically for online poker tournaments combined with some simple heuristics. This bot showed a profit of \$26.8 after over 200 \$1.2 games which translates to a 11% return on investment.

In conclusion it can be said that even though there exists a demand for poker bots there are some difficulties in selling such software. First of all most casinos do not allow poker bots to be used, and second if too many players would use automated poker agents the pleasure of recreational players could be in danger. There might however be a market for a new type of casino, aimed specifically at players using poker bots. However, even if there are no direct business opportunities for the use of poker bots, advances in the techniques used in this area might prove to be useful in other domains.

After reviewing the papers studied for this research it can be concluded that there is still room for improvement and that there exist open issues and subjects.

From the experiments it can be concluded that a simple rule-based system suffices for competing at low buy-in on-line poker tournaments.

Preface.....	2
Summary.....	3
1. Introduction.....	5
2. The game of poker.....	6
3. Available software.....	9
4. Main approaches.....	11
5. Experiments.....	18
6. Results.....	20
7. Conclusion.....	22
8. References.....	23

## 1. INTRODUCTION

Poker is in many ways an interesting game to study in the context of AI. There is uncertainty, incomplete information, deception, a clear objective, one or more competing agents and a set of well defined rules. All of which are interesting subjects in AI.

The main goal of this paper is to give an overview of existing solutions and to get hands on experience with automated play in low stakes internet games.

Playing poker shows much resemblance with the trading on financial markets, much of the same aspects are shared in both domains.

Trading software for the financial markets has been in use for many years mostly by big financial institutions that exploit arbitrary investment opportunities. Such software makes trading decisions based on the information available. The same approach can be used in poker.

This paper discusses the software and algorithms that could be used to achieve this task. What are the possibilities and limitations of existing software? Which algorithms and techniques are applicable to poker? Is there much space for improvement?

The paper is organized as follows. First, a brief description of the rules of poker is given. Next an overview is given of the most important readily available packages in this area. Then four main approaches that could be applied to poker are discussed. Next, an own experiment is described where two knowledge based bots are tested in low stakes internet poker tournaments. Finally, In the end of the paper a conclusion is given.

## 2. THE GAME OF POKER

This paper focuses on the currently most popular form of poker: Texas Hold 'em <sup>1</sup>. This game can be played with two or more players and consists of four betting rounds. These rounds are called: **pre-flop**, **flop**, **turn** and **river**. The flop turn and river are also commonly referred to as **post-flop**. In every round a player can choose between the actions listed in table 1. To be able to continue to the following round all players must have put in the amount of the current bet in the pot. The betting structure of the game can either be fixed limit, pot limit or no limit. In fixed limit the amount a player can bet is set fixed for each betting round. In pot limit the maximum amount is restricted by the current size of the pot and in no limit a player can bet up to the size of his chip stack at that point.

Action	Description
Fold	Give up and throw away cards.
Check	Do nothing and pass the action to the next player when there is no current bet.
Call	Put in the amount of the current bet.
Bet	Place a bet in the pot when no other players have done so and thus setting a price for the other players to remain in the hand.
Raise	Raise the amount of the current bet and put this amount in the pot.

Table 1: Possible poker actions

---

<sup>1</sup> [http://www.poker.com/rules\\_texas\\_holdem.htm](http://www.poker.com/rules_texas_holdem.htm)

### 2.1. PRE-FLOP

Before any cards are dealt there are two forced bets which are called the **small blind** and **big blind**. The order in which players have to post the blinds is determined by their position. This position is shifted every new hand so that every player has to be small and big blind at some point. The reason for these forced bets is to create an initial pot. With an empty pot there would be no reason for any further betting.

After the blind money is put in the pot all players are dealt two cards face down which are called their **hole cards**. The choice of action to take is mostly based on the strength and potential of these hole cards in combination with the position of the player. A player who is first to act has a disadvantage compared to a player who is last to act because the later can see what players in front of him have done.

### 2.2. POST-FLOP





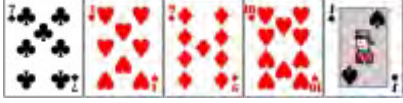




On the flop three **community cards** are dealt faced up. All players still in the pot can have combinations with these cards in the middle, such as a pair of aces when a player holds an ace and an ace is dealt on the flop. The actions taken by players at this point are mainly based on the strength of the combination or the potential to improve on further rounds. A hand with potential to improve is called a **draw**. Players can also **bluff** at this point which is the action of betting or raising without any great hand strength.

On both the turn and the river one more additional card is dealt and the actions that can be taken on both rounds are similar to those on the flop.

A player can win the pot in two ways: by betting or raising in any of these rounds such that all players fold, or by having the best five card combination after the last betting rounds.

### 2.3. HAND COMBINATIONS

The possible hand combinations in Texas Hold 'em are as follows, listed from weakest to strongest<sup>2</sup>.

Hand	Example
No pair / High card	
One pair	
Two pair	
Three of a kind	
Straight (5 consecutive ranked cards)	
Flush (5 cards of the same suit)	
Full house (3 of kind and a pair)	
Four of a kind	
Straight flush (5 consecutive ranked cards of the same suit)	

If two players hold the same combination, one with the highest card wins (for example a straight with the 7 as a highest card vs a straight with a 9 as the highest card) or in case the combination is exactly the same the players split the pot. For a more detailed description of the rules and strategies of poker the reader is referred to (Sklansky 1999).

<sup>2</sup> [http://en.wikipedia.org/wiki/Poker\\_probability](http://en.wikipedia.org/wiki/Poker_probability)



### 3. AVAILABLE SOFTWARE

There exists a big market for poker software. A simple search on Google for the term “poker software” results in about 7 million hits. The purpose of these tools is to improve the quality of a poker player’s strategy. Poker software can be divided in a couple of main groups: statistical software which provides insight in the behavior of poker players, offline training software, advisory software which will give advice during play, and various calculators. All this software is aimed at the roughly 1.8 million players that play everyday generating an income of \$2.4 billion a year for the various online casinos (Newsweek 2005).

The market for poker bots that can play poker fully automatically is one with a big demand and only a small supply of quality software available. The business potential of this market is however questionable as most casinos try to prevent players from using software to automate play. If too many players would use this software casinos might take better action in preventing players to use such software.

The main software available for automated poker play is described in more detail below. A set of rules and formulas for automated play is usually referred to as a profile in the context of poker software, not to be confused by opponent profiling.

#### 3.1. ONLINE HOLDEM INSPECTOR (OHI) <sup>3</sup>

This program is in itself no actual poker bot in the sense that it cannot play poker automatically. An advice of the course of action to take by the player is given based on an extensive profile that can be set in the program. Complete profiles are being sold by various third parties. Such a profile basically contains a collection of rules. OHI can also make use of opponent statistics collected with poker tracker, a popular statistical poker tool.

There are third party applications available that will automatically execute the recommended action from OHI. There is also a market for OHI profiles.

#### 3.2. HOLDEM BOT <sup>4</sup>

Holdem bot is a similar program to OHI only with a less extensive interface. It is however a complete botting solution that is compatible with various poker rooms. The program is shipped with some profiles that are claimed to make profit.

Just like OHI, Holdem bot is rule based only without possibility to use opponent statistics.

---

<sup>3</sup> <http://www.pokerinspector.com/>

<sup>4</sup> <http://www.bonusbots.com/holdempokerbot.htm>

### 3.3. OPEN HOLDEM (OH)<sup>5</sup>

OH is an open source framework that provides everything needed to program a poker bot. OH derives the game state information by capturing the graphics of the poker client. OH comes with tools to create a map which tells how to interpret the graphics. This map can then be used by the main program to 'connect' to the poker client. The main program supports scripts in a native C like language or perl to make rule based bots. For more complicated solutions OH can be interfaced with an external module which for example can be written in C++. This makes the possibilities with OH virtually unlimited. The main difference with other solutions is that OH is more geared towards software developers..

OH is an open source counterpart of the commercial Win Holdem software. Since OH was created many Win Holdem users switched over because OH is backwards compatible with Win Holdem.

### 3.4. POKER ACADEMY (PA)

PA is a poker training toolbox that offers some of the most sophisticated bots available created by a poker research group of the University of Alberta. However, the software offers no direct support for enabling these bots play in on-line poker games. PA comes with a java interface for bot developers to let their bots play against bots available in the program.

Some of the bots in question and their techniques are discussed in more detail in the next chapter.

### 3.5. TURBO HOLDEM<sup>6</sup>

Turbo holdem is a package similar to PA in that it is also a training program. The program is available for various types of poker and is based on a fixed rule based engine. However it is mentioned in (Billings 2006) that this is system is easily beaten by capable human players and that such a fixed rule based approach is not suitable for poker.

### 3.6. VARIOUS OTHERS

There are various other complete botting solutions offered for sale on the internet. Most of them are very simple programs with few settings which are claimed to be profitable in the long term. It is however the consensus in the poker botting community that these are scams. If their software would be real long term winners the authors of such programs would be better off by keeping their program for themselves and make money that way. In conclusion, it can be said that anyone who wants a winning poker bot should program it himself..

---

<sup>5</sup> <http://code.google.com/p/openholdembot/>

<sup>6</sup> <http://wilsonsw.com/texas-holdem-poker/index.html>

## 4. MAIN APPROACHES

In (Watson, Rubin 2011) an overview is given of algorithms applied to poker. This paper was used as a main source for determining which algorithms were interesting to describe in this chapter.

This chapter first discusses knowledge-based approaches which can be seen as the first generation of poker AI. Next "on the fly" simulation is discussed. Then game theoretic approaches are discussed and this chapter is concluded with a description of an application of case based reasoning to poker.

### 4.1. KNOWLEDGE-BASED

In the knowledge-based approach specific domain knowledge of poker is encoded in a bot. This is the most simple and straightforward approach. The solutions intended for online poker play mentioned in the previous chapter all belong to this class of bots. For a general introduction to knowledge-based systems the reader is referred to (Stefik 1995).

A further distinction in knowledge based algorithms can be made between rule-based and formula-based algorithms.

The rule-based approach mainly consists of a collection of if-then statements for specific situations encountered. The formula-based approach is slightly more elegant. Based on some input which represents a summary of the current game state the action to take is calculated.

Rule-based systems can either be fixed like the earlier mentioned Turbo Holdem, or probabilistic, where a rule corresponds to a set of weights for actions to be taken. A random number is then drawn to determine which action to take.

Examples of rule-based approaches to poker can be also be found in the poker literature. (Sklansky 2007) describes a rule based system for tournament poker (an area of poker where little research is done for the application of AI).

The described rule-based approach is called the "System". This system was originally designed for a novice in poker who wanted to enter a no-limit hold 'em poker tournament. The system only allowed for two moves in the pre-flop round: moving all-in (raising with all chips) or folding. By doing this, the game is in effect reduced to a much simpler game with only two moves and one round of betting (after a player is all-in that player has no further possible actions to take). The effectiveness of the system lied in the aversion of (capable) poker players of betting most of or their whole stack pre-flop.

The initial version of the system was actually so simple that it only consisted of two rules:

1. If someone else has raised in front of you, move in all your chips with aces, kings or ace-king suited. Otherwise fold.
2. If no one else has raised in front of you, move in all your chips with any pair, any ace-other suited, ace-king (suited or off suit), or two suited connected cards except for four-trey or trey-deuce.

An improved version of the system was also described which is a formula-based version of the system. The principle remains the same, a player either moves in or folds pre-flop. The formula is used to calculate the odds a player is laying by moving all-in and discounts this value by the number of players yet to act and already in the pot. The resulting number, called the key number (4.1) is then used in conjunction with a table listing the hands to move-in with combined with key number intervals.

$$((\text{Stack size}) / (\text{Blinds} + \text{Antes})) * (\# \text{ players to act}) * (\# \text{ callers in front} + 1) \quad (4.1)$$

An example of an implementation of a bot that uses a knowledge-base is Poki (Billings 2006) which is available as an opponent in Poker Academy. Poki is designed to play fixed limit poker against multiple opponents.

The pre- and post-flop strategies in this bot are completely separated. The reason for this was that in the pre-flop round there is little information available to base decisions on.

The rules for the pre-flop betting were made by running an offline Monte-Carlo simulation of all possible hands a player could receive. The simulation consisted of all players calling pre-flop without any additional betting in later rounds and computing the percentages that each hand wins. The resulting ranking of hands was closely correlated to hand rankings found in poker literature.

Hand strengths after the pre-flop betting round are a little less straightforward to calculate. A pair of kings for example has less value when an Ace is dealt on the flop.

Three measures were used for the post flop rounds, *effective hand strength*(EHS) the *hand strength*(HS) and the *positive potential*(PPot).

The HS is calculated by on-line enumerating all possible hands an opponent could have and count wins(+1) and ties(+1/2). The total count is then divided by the number of hands and this percentage gives the HS. When facing  $n$  opponents this value is raised to the power  $n$ .

Simply enumerating all possible hands is unrealistic as it is for example more likely for a typical opponent to hold a pair of nines than it is to hold a 3 and a 2. To overcome this, a weight table is maintained in memory for each opponent. This weight table contains all possible two cards an opponent can hold with a weight between 0 and 1 assigned to it. Initially all weights are set to 1 and are updated as the game progresses and hands are observed.

Ppot is a measure of how likely it is for a hand that is not currently best to improve to the best hand. A player missing one card to complete a strong hand with the turn and river to come would result in a high Ppot. The value is calculated similar to the HS but in this case enumerating over all possible hands to come in future rounds and calculating the percentage of times a player has the best hand in the last round.

The EHS is then calculated by (4.2)

$$EHS = HS^n + (1 - HS^n) \times P_{pot} \quad (4.2)$$

where  $n$  is the number of opponents. The EHS can be interpreted as the current strength of the hand + the probability the hand is not the best hand at the moment, multiplied by its potential.

Together with the game context, the EHS is used by the rule-base of the system to produce a probability triple of the form  $\{pr(\text{fold}), pr(\text{call}), pr(\text{raise})\}$  which assigns a probability to each possible action to take. One of these actions is then chosen at random which ensures a degree of unpredictability which is important in solid poker play.

The exact nature of the rules is unfortunately not disclosed in the paper where Poki is described because according to the author the exact nature of the rules were of little scientific relevance.

An example of a rule-based bot for no limit hold 'em is described in (Mccurley 2009). This bot implemented a pre-flop strategy found in the poker literature and used rules that were based on similar metrics as mentioned above. This bot was tested on an online poker site for 545 hands which resulted in a loss of 170 big blinds.

## 4.2. MONTE-CARLO SIMULATION

Monte-Carlo simulation is a statistical method of obtaining an approximation of an outcome by repeated statistical sampling. In general the accuracy of the outcome increases as the sample size is increased.

In the description of Poki it was already discussed how simulation can be used to obtain various hand strength measures. A more advanced application of simulation in poker is to simulate various scenarios a hand can play out calculating an expected value for these scenarios.

The strength of simulation is that it can be used in situations where exhaustive search through the whole game tree is infeasible which is generally the case in poker.

Later versions of Poki used a simulation based component to come to a betting decision. It does however make use of the rule-based component to 'predict' its own future decisions when simulating. The simulation component of Poki plays out many likely scenarios while keeping track of the money won or lost for each betting decision. For each scenario two simulations are run, one for calling or checking (passive) and one for betting or raising (aggressive). To simulate an opponent move the statistical opponent model described earlier is used. When the simulation is complete the action is taken with the greatest expected value.

A great improvement in using simulation instead of rules is that its more adaptive and capable in uncovering subtle strategies such as deceptive plays. It is also less reliant on expert knowledge as possibly a simpler rule base could be used instead of the extensive rule base used in Poki for the simulated action to take.

Another successful bot based on simulation is AKI-Realbot which is described in (Schweizer et al 2008). This bot placed 2<sup>nd</sup> in the AAAI-08 <sup>7</sup>Computer poker challenge which was a competition for poker bots competing in 6 handed fixed limit hold 'em.

The simulation process was more extensive than in Poki in the sense that it simulated all available future actions it could take instead of finding a likely action based on rules. For predicting opponent actions and likely hands it used straightforward opponent statistics collected during play.

The main reason for AKI-Realbos success lies in the post processing step of the simulation results. For each opponent the bot kept record of how much money was won or lost to this opponent. It is also determined by statistics if an opponent is weak which means that an opponent folds too much or calls too often after the flop. These weaknesses were exploited by discounting the expected value of folding (not playing a hand) with a delta value. This delta value would be higher for weaker opponents which results in a smaller discounted expected value for folding which in turn results in the bot choosing to call or raise.

---

<sup>7</sup> <http://www.computerpokercompetition.org/>

### 4.3. GAME THEORY

Game theory is defined as “the formal study of decision-making where several players must make choices that potentially affect the interests of the other players” (T. L. Turocy, B. von Stengel 2001).

An important result of game theory is that for every finite game with finite strategy sets there exists an equilibrium strategy. Such a strategy has the property that when all players implement it and one of the players would deviate this player would incur an expected loss.

To illustrate how game theory can be used to find an equilibrium strategy the game of rock, paper, scissors (RPS) will be analyzed. In this game two players simultaneously choose either rock, paper or scissors where rock beats scissors, paper beats rock and scissors beats paper. The possible choices with their corresponding payoffs can be represented in a matrix which in game theory is called *normal form*. This matrix lists all possible payoffs for each end state of the game.

<i>Player A / Player B</i>	<i>Rock</i>	<i>Paper</i>	<i>Scissors</i>
<i>Rock</i>	0	-1	1
<i>Paper</i>	1	0	-1
<i>Scissors</i>	-1	1	0

It can directly be seen from the table that when player A choose Paper and player B choose Rock a payoff of 1 for player A would be the result. RPS is an example of what is called a zero sum game which means that the sums of all payoffs is 0 and thus a payoff of 1 for player A implies a payoff of -1 for player B.

To find a game theoretic solution to the game the minimal payoff of player A could be maximized. The solution would consist of a triple of probabilities for the choice for player A. In game theory such a triple or more general n-tuple, is called a mixed strategy. It turns out that the problem of finding a solution can be solved by means of a linear programming problem, where a linear objective function is maximized under some linear constraints. In case of the RPS example this is done as follows.

Let all  $X_i$  correspond to probabilities that player A chooses a specific action with:

$X_1$  = player A chooses Rock,  $X_2$  = player A chooses Paper and  $X_3$  = player A chooses Scissors.

Let  $W = \min\{X_2 - X_3, -X_1 + X_3, X_1 - X_2\}$

Where  $W$  is an expression for the minimal payoff of player A. The minimal payoff is chosen because we want a solution that results in the highest payoff assuming that player B has a strategy such that he is minimizing our payoff.

The minimization of  $W$  can be written as a set of constraints and the resulting LP-problem is given by:

Maximize  $W$

Under:

$$W \leq X_2 - X_3,$$

$$W \leq -X_1 + X_3$$

$$W \leq X_1 - X_2$$

$$X_1 + X_2 + X_3 = 1$$

$$X_1, X_2, X_3 \geq 0$$

The resulting solution is  $X_1, X_2, X_3 = 1/3$ .

The example of RPS is of course extremely simple compared to strategic rich games such as poker.

The number of possible game states for example in 2 player limit hold 'em is  $10^{18}$  which makes it infeasible to solve it in the same way as RPS which has only 9 game states.

A first step is to represent the game in a different way. This can be done by using the sequence form which represents strategies as probabilities which can be taken at each of the nodes in the game tree which reduces the size of the matrix drastically.

Even after switching to sequence form the game is still too large to be solved and a number of abstractions can be made to reduce the size of game. Solving the abstracted game will lead to a so called near equilibrium strategy which is an approximation to the true equilibrium strategy. In practice this means that the resulting strategy has vulnerabilities which can be exploited.

An intuitive abstraction is to group the two card combinations that a player can hold together in a group which is known as bucketing. This can be done by a domain expert, or dynamically by means of running simulations to obtain hand strengths and group cards with similar strengths together.

Other abstractions that can be made are a reduction in the number of betting rounds, for example by ignoring the last round or grouping two rounds together as one.



#### 4.4. CASE BASED REASONING

Case based reasoning (CBR) is a very intuitive learning method. To classify an unknown instance, a so-called case base is referred to which holds known cases with their classifications. A distance measure combined with weights for the attributes determines which cases are similar and the unknown case is classified according to similar cases. The case base is then updated with the new case. For a more detailed discussion of CBR the reader is referred to (Mitchell, 1997)

In (Sandven, Tessem 2006) an experiment is described where this technique is applied to fixed limit holdem poker for different number of opponents. In general CBR has not been applied much to games. One of the reasons is that for most games very large case bases are required.

When the bot faces a decision it consults its case base for similar cases, if not enough cases were found a random play is made and the result is stored in the case base. Initially the case base is empty and the bot makes random plays. As the case base grows more similar cases are found and the number of random plays decreases.

To fill the case base a large number of hands were played against bots found in Poker Academy.

The similarity between cases was measured by comparing the following attributes: a measure for hand strength, position, number of opponents and number of bets to call.

Besides these features so called unindexed indices are stored in the case base. These variables consisted of the strategy chosen, the investment made, the size of the pot and the result of the strategy.

For the pre-flop round and post-flop rounds two separate case bases were made because the post-flop contains some different variables such as the potential of the hand given the community cards dealt and the relative position which is the position relative to the last player to act in the current round.

To make a decision the case base is consulted and if sufficient similar cases are found the play with the best result is chosen. To test the bots ability it was tested against a rule-based bot from Poker Academy and the results show that it was most successful against three opponents, showing a profit after 50000 hands. Against six or eight opponents the bot is however not able to show a profit.

Another attempt at creating a bot using CBR is described in (Watson and J. Rubin 2008).

The case base in that experiment was built by letting bots from poker Academy play against each other.

When retrieving a case, a probability triple is constructed for the three possible actions (fold, call, raise). The probabilities directly correspond to the frequencies of the actions in the case base. A decision is then made by selecting one of the actions at random.

The bot was tested against players in online poker rooms, first against opponents on play money table which resulted in a profit. On the real money tables, however, the bot was not able to win.

Using a different case base, such as the hand history of a skilled human player might result in very different results and would be an interesting experiment.

## 5. EXPERIMENTS

This chapter gives an overview of some experiments of using a bot to play online double or nothing (DoN) sit and go poker tournaments. This type of poker tournament consists of ten players where the five players remaining in the end receive double their buy in minus a fee paid to the casino. Because every player receives the same price amount, there is little incentive to accumulate a large number of chips. The main goal of the tournament is survival until five players are remaining although, many players are not aware of this.

For the experiments the lowest buy in was chosen which is \$1.2, 20 cents of that goes to the casino and the rest to the prize pool so that the winning players all receive \$2 resulting in a net win of 80 cents.

The tournament starts out with a small big blind compared to the stack sizes of the player. Every player gets 1500 chips and the big blind is 20. The size of the big blind determines the minimal amount a player has to pay in order to play a hand. Every 8 minutes the big blind is increased, this will eventually cause the stack sizes to be small compared to the blind. At this point the game is mostly played before the flop as there is little room for betting post-flop with such a small stack size. The bots designed in the experiments do most of their work in this phase of the tournament.

This form of poker is chosen because little results were found in the literature concerning tournament poker bots and because the author of this paper is well acquainted and successful in this form of poker.

The goal of the experiments is to explore the process of implementing an agent using a knowledge-based approach, and to investigate how successful this agent performs against the weak opposition found in low buy-in internet poker tournaments.

The experiments were conducted using Open Holdem (OH) combined with AutoIT scripts to automate the task of opening and closing tournaments to achieve complete automated play without requiring supervision.

### 5.1. TABLE INTERACTION

The first step of the experiment was to achieve successful interaction with the poker client software. To this end a table map had to be created for the casino at which the experiments were performed. To do this OH was used to create a sufficient number of screen shots during play. The screen shots are then used to define regions, fonts and buttons on the poker client. Together with some meta data such as the title bar text and window size this forms the table map.

Getting a first working version of the table map took approximately a day of work. During further experiments some bugs in the mapping were found which were mostly due to undefined fonts and were easily corrected.

## 5.2. SLANKSY DoN BOT

For the first experiments a bot was created based on the simple formula based system explained in the previous chapter. The resulting bot performed only two actions: it either folds its cards or it goes all-in (betting all of one's chips).

When the system would be used as is, the bot would get into situations where it would bet all its chips early on in the tournament. This behavior is very unwanted, especially at low buy-in tournaments because the experience is that by simply waiting, other players will eliminate each other.

To this end some specific heuristics were added which instructs the bot to only move all-in with two aces or kings when the bot has a stack greater than 10 big blinds and employ the system when the stack size drops below 10 big blinds. Effectively this will make the bot fold most of its cards in the first half hour of the tournament. By doing this it already gains because at this point there are already some players eliminated from the tournament most of the time.

When the bot moves all-in, other players are forced to fold their cards unless they happen to have a strong hand or very large stack size which enables them to gamble a large part of their stack without risking elimination. If all players fold this will result in winning the amount of the blinds, if one or more of the players call, the post flop cards will be dealt. In this case, the bot will either win a big pot when it has the best hand when all cards are dealt or it will be eliminated from the tournament.

## 5.3. MOSHMAN DoN BOT

For the second experiment the formula from the first experiment was replaced by a simple table found in (Moshman 2006). This table lists for every position and stack size cards to go all-in with in the later stages of a tournament when the blinds are high. It is noted that the table should serve as a rough guideline and is based on experience and expected value calculations.

The maximum stack size listed in the table is 10 big blinds, which causes the bot to wait until the later stages of the tournament in the same way as the Sklansky bot.

In addition, some refinements were made to the heuristics to make it more aware of its 'stack health' compared to other players. An example of a rule which was added states that no action should be taken when 6 players are remaining and the bot is ranked fourth or higher when looking at the stack sizes of all remaining players. This in effect prevents the bot from taking any action when only one player has to be eliminated and there are two likely candidates for elimination.

Another addition made was making the bots play paired cards early on in the tournaments. The bot simply calls pre-flop with paired cards when no one has raised. The bot will then simply move all-in when a third card of the same kind is dealt on the flop giving the bot a three of a kind which is often the best hand.

## 6. RESULTS

### 6.1. SLANKSY DoN BOT

At first the bot seemed to do quite well, however by inspecting its playing style the bot was often either too aggressive or too passive. It was also often not aware of situations where waiting until the elimination of one or two players who had very little chips left was clearly the best play. Another factor contributing to the poor results might lie in the lack of aversion of low stake players to gamble all or most of their chips. This results in other players calling too often when the bot has moved all-in and thus creating a situation of directly risking elimination.

The results after letting this bot play 50 tournaments can be seen in figure 1.



Figure 1: Results of the Sklansky DoN bot

## 6.2. MOSHMAN DoN BOT

The results of the Moshman DoN bot were very promising after over 200 were tournaments played as can be seen in figure 2.

Tournament players express their ability often in terms of return on investment.

Calculating the return on investment based on the experimental results gives:

$$\text{net profit} / (\text{number of tournaments} * \text{buy-in amount}) = 26.8 / 247,2 = 11\%$$

Which is an excellent result for double or nothing tournaments. It must however be noted that 200 tournaments is still a relatively small number of tournaments due to the variance involved in poker. Also the presence of an excessive win streak of 16 tournaments probably causes the result to be higher than its actual long term return on investment. It is nevertheless safe to say that a heuristics based system suffices to make a profit at micro stakes online poker.

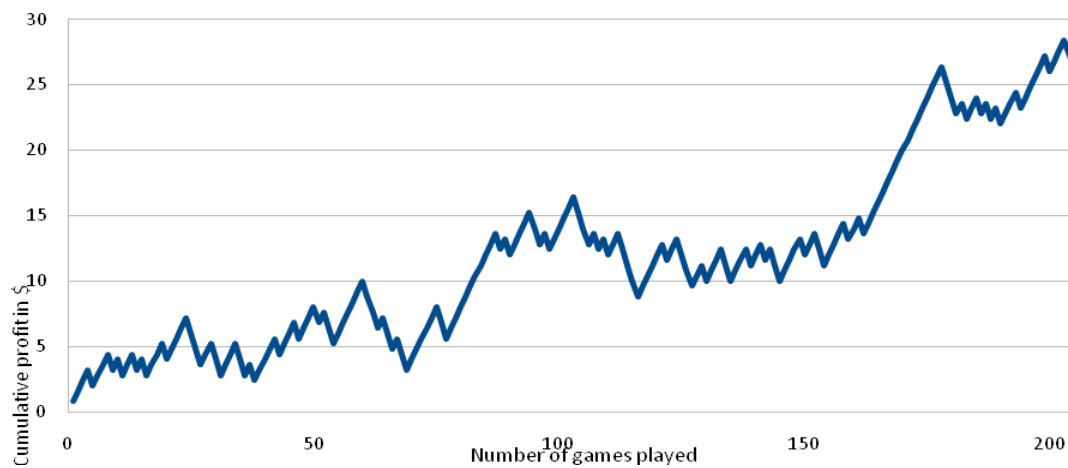


Figure 2: Results of the Moshman DoN bot

## 7. CONCLUSION

The main goals of the paper were to review existing approaches of applying AI in poker and to get some hands on experience with automated play.

The market for poker software that can play fully automatically is one with a big demand and a small supply. It is also a difficult market because most casinos do not allow automated play. Furthermore, if on-line poker would be dominated by automated players the pleasure of the recreational players would be ruined. There might exist an opportunity for a new type of poker rooms specially aimed at players using bots. However, even there are no direct business opportunities for the use of poker bots, advances in the techniques used in this area might prove to be useful in other domains.

After reviewing the papers studied for this research it can be concluded that there are still open issues and subjects. For example currently it is not possible to find a true equilibrium strategy for poker due to the computational complexity. In most papers AI in fixed limit poker is discussed while tournament poker and no limit poker are subjects where little research has been done. Especially no limit hold 'em would be an interesting subject because of the increased complexity involved due to the unrestricted betting format. There is also room for improvement of existing approaches and research is still ongoing.

The aim of the experiments done for this paper differed from the experiments done in the papers that were reviewed. Instead of trying to create a bot capable of competing with strong opposition, it was chosen to create a bot which exploits tendencies of weak players. It was demonstrated that this can be done by implementing a simple set of rules based on experience and domain knowledge.

## 8. REFERENCES

- D. Billings (2006), *Algorithms and assessment in computer poker*, Ph.D Dissertation University of Alberta
- P. Mccurley (2009), *An Artificial Intelligence Agent for Texas Hold'em Poker*, Undergraduate dissertation University of Newcastle Upon Tyne
- T. Mitchell (1997), *Machine Learning*, McGraw-Hill (chapter 8)
- C. Moshman (2007), *Sit'n Go strategy*, two plus two (page 247-250)
- A. Sandven and B. Tessem (2006), *A Case-Based Learner for Poker*, The Ninth Scandinavian Conference on Artificial Intelligence (SCAI)
- I. Schweizer et al (2009) *An exploitative Monte-Carlo Poker Agent*, Springer-Verlag
- D. Sklansky (1999), *Hold' em poker for advanced players*, two plus two publishing
- D. Sklansky (2007), *Tournament poker for advanced players*, two plus two publishing
- M. Stefik (1995), *Introduction to knowledge systems*, Morgan Kaufmann Publishers
- T. L. Turocy, B. von Stengel (2001), *Game Theory (draft of an itroductory survey of game theory)*, Academic Pres
- Newsweek (August 14, 2005), *Going all in for online poker*, The Daily Beast Newsweek Publishing
- I. Watson and J. Rubin (2008), *Capser a Case-Based Poker-Bot*, Spring-Verlag
- I. Watson and J. Rubin (2011), *Computer poker: A review*, Elsevier