CASPER: a Case-Based Poker-Bot

Ian Watson and Jonathan Rubin

Department of Computer Science
University of Auckland, New Zealand
ian@cs.auckland.ac.nz, jrubin01@gmail.com

Abstract. This paper investigates the use of the case-based reasoning methodology applied to the game of Texas hold'em poker. The development of a CASe-based Poker playER (CASPER) is described. CASPER uses knowledge of previous poker scenarios to inform its betting decisions. CASPER improves upon previous case-based reasoning approaches to poker and is able to play evenly against the University of Alberta's Pokibots and Simbots, from which it acquired its case-bases and updates previously published research by showing that CASPER plays profitably against human online competitors for play money. However, against online players for real money CASPER is not profitable. The reasons for this are briefly discussed.

Key words: Case-Based Reasoning, Game AI, Poker

1. Introduction

The game of poker provides an interesting environment to investigate how to handle uncertain knowledge and issues of chance and deception in hostile environments. Games in general offer a well suited domain for investigation and experimentation due to the fact that a game is usually composed of several well defined rules that players must adhere to. Most games have precise goals and objectives that players must met to succeed. For a large majority of games the rules imposed are quite simple, yet the game play itself involves a large number of very complex strategies. Success can easily be measured by factors such as the amount of games won, the ability to beat certain opponents or, as in the game of poker, the amount of money won.

Up until recently AI research has mainly focused on games such as chess, checkers and backgammon. These are examples of games that contain perfect information. The entire state of the game is accessible by both players at any point in the game, e.g. both players can look down upon the board and see all the information they need to make their playing decisions. These types of games have achieved their success through the use of fast hardware processing speeds, selective search and effective evaluation functions [1]. Games such as poker on the other hand are classified as stochastic, imperfect information games. The game involves elements of chance (the actual cards that are dealt) and hidden information in the form of other player's hole cards (cards that only they can see). This ensures that players now need to make decisions with uncertain information.

The focus of this paper is to investigate the application of Case-Based Reasoning (CBR) to the game of poker. We have developed a poker-bot, called

CASPER (CASe-based Poker playER), that uses knowledge of past poker experiences to make betting decisions. CASPER plays the variation of the game known as "limit Texas Hold'em" and has been tested against other poker bots and real players.

2. Related Research

Recently there has been a dramatic increase in the popularity of the game of Texas hold'em. This popularity has also sparked an interest in the AI community with increased attempts to construct poker robots (or bots). Recent approaches to poker research can be classified into three broad categories:

1. **Heuristic rule-based systems**: that use knowledge such as the cards a player holds and the amount of money being wagered, to inform a betting strategy.
2. **Simulation/Enumeration-based approaches**: that consist of playing out many scenarios from a certain point in the hand and obtaining the expected value of different decisions.
3. **Game-theoretic solutions**: that attempt to produce optimal strategies by constructing a game tree.

The University of Alberta Poker Research Group currently lead the way having investigated all of the above approaches. The best known outcome of their efforts are the poker bots nicknamed Loki/Poki [2] & [3]. Loki originally used expert defined rules to inform a betting decision. While expert defined rule-based systems can produce poker programs of reasonable quality [3], various limitations are also present. As with any knowledge-based system a domain expert is required to provide the rules for the system. In a strategically complex game such as Texas hold'em it becomes impossible to write rules for all the scenarios that can occur. Moreover, given the dynamic, nondeterministic structure of the game any rigid rule-based system is unable to exploit weak opposition and is likely to be exploited by any opposition with a reasonable degree of strength. Finally, any additions to a rule-based system of moderate size become difficult to implement and test [4]

Loki was redeveloped as Poki [5]; a simulation-based betting strategy was developed that consisted of playing out many scenarios from a certain point in the hand and obtaining the expected value of different decisions. A simulation-based betting strategy is analogous to selective search in perfect information games. Both rule-based and simulation-based versions of Poki have been tested by playing real opponents on an IRC poker server. Poki played in both low limit and higher limit games. Poki performed well in the lower and higher limit games [3]. More recently The University of Alberta Computer Poker Research Group have attempted to apply game-theoretic analysis to full-scale, two-player poker. The result is a poker bot known as PsOpti that is "able to defeat strong human players and be competitive against world-class opponents" [6].

There have been several other contributions to poker research outside of Alberta: Sklansky and Malmuth have detailed various heuristics for different

stages of play in the game of Texas hold'em [7] & [8]. The purpose of these rules, however, has been to guide human players who are looking to improve their game rather than the construction of a computerised expert system. Korb produced a Bayesian Poker Program to play five-card stud [9], whilst Dahl investigated the use of reinforcement learning for neural net-based agents playing a simplified version of Texas hold'em [10].

We have encountered few attempts to apply the principles and techniques of CBR to the game of poker. Sandven & Tessem constructed a case-based learner for Texas hold'em called Casey [11]. Casey began with no poker knowledge and builds up a case-base for all hands that it plays. They report that Casey plays on a par with a simple rule-based system against three opponents, but loses when it faces more opponents. Salim & Rohwer have attempted to apply CBR to the area of opponent modeling [12], i.e., trying to predict the hand strength of an opponent given how that opponent has been observed playing in the past.

3. Texas Hold'em

Texas hold'em is the variation used to determine the annual World Champion at the World Series of Poker. This version of the game is the most strategically complex and provides a better skill-to-luck ratio than other versions of the game [8]. Each hand is played in four stages: the preflop, flop, turn and the river. During each round all active players need to make a betting decision to fold, call or raise. The complete rules of Texas Hold'em can be found online at: http://en.wikipedia.org/wiki/Texas_hold_'em

4. System Overview

CASPER uses CBR to make a betting decision. This means that when it is CASPER's turn to act it evaluates the current state of the game and constructs a target case to represent this information. A target case is composed of several features recording important game information such as: CASPER's hand strength, how many opponents are in the pot, how many opponents still need to act and how much money is in the pot. Once a target case has been constructed CASPER then consults its case-base to try and find similar scenarios from the past. CASPER's case-base is made up of a collection of cases composed of their own feature values and the action that was taken, i.e. fold, check/call or bet/raise. CASPER uses the k-nearest neighbour algorithm to search the case-base and find the most relevant cases, these are then used to decide what action should be taken.

CASPER was implemented using Poker Academy Pro 2.5 and the Meerkat API. The University of Alberta Poker Research Group provides various poker bots with the software including instantiations of Pokibot and the simulation based Simbot. These poker bots were used to generate poker cases for CASPER. Note that our intention was not to simulate Pokibot or Simbot. We used these bots merely to generate poker cases, since unlike chess or bridge there are no

libraries of poker games available. In particular we wanted to see if it was possible to build a competitive poker bot with little or no knowledge engineering. Initially 7,000 hands were played between the poker bots and each betting decision witnessed was recorded as a single case in CASPER's case-base. Alberta's bots have proven to be profitable against human competition in the past [5], so we hypothesise that the cases we obtained are of sufficient quality to enable CASPER to play a competitive game of poker against people.

4.1 Case Representation & Retrieval

CASPER searches a separate case-base for each seprate stage of a poker hand (i.e. preflop, flop, turn and river). The features that make up a case and describe the state of the game are described in more detail in [13]. Each case has a single outcome that is the betting decision that was made. Once a target case has been constructed CASPER retrieve the most similar cases it has from its case-base. The k-nearest neighbour algorithm is used to compute a similarity value for all cases in the case-base. Two separate similarity metrics are used depending on the type of feature [13].

After computing a similarity value for each case in the case-base a descending quick sort of all cases is performed. The actions of all cases that exceed a similarity threshold of 97% are recorded. Each action is summed and then divided by the total number of similar cases that results in the construction of a probability triple (f, c, r) that gives the probability of folding, calling or raising in the current situation. If no cases exceed the similarity threshold then the top 20 similar cases are used. As an example, assume CASPER looks at its hole cards and sees A♥ and A♠. After a search of its preflop case-base the following probability triple is generated: (0.0, 0.1, 0.9). This indicates that given the current situation CASPER should never fold this hand, he should just call the small bet 10% of the time and he should raise 90% of the time. A betting decision is then probabilistically made using the triple that was generated (i.e., raise).

5. Results

CASPER was initially evaluated at two separate poker tables. The first table consisted of strong, adaptive poker bots that model their opponents and try to exploit weaknesses. As CASPER has no adaptive qualities of its own it was also tested against non-adaptive, but loose/aggressive opponents. The results from these experiments are reported in detail in [13]. To summarise, the best paramaterised version of CASPER made a profit of +0.03 small bets per hand, or $0.30 for each hand played.

5.1 Real Opponents - Play Money

CASPER was then tested against real opponents by playing on the 'play money' tables of internet poker websites. Here players can participate in a game of poker using a bankroll of play money beginning with a starting bankroll of $1000. All

games played at the 'play money' table were $10/$20 limit games. At each table a minimum of two players and a maximum of nine players could participate in a game of poker. CASPER was tested by playing anywhere between one opponent all the way up to eight opponents. Figure 1 displays the results for CASPER (using hand picked feature weights) and CASPERGeneral using feature weights derived by an evolutionary algorithm (to be described in a future paper), as well as a random opponent that makes random decisions (used as a baseline comparison).

The results suggest that the use of CASPER with hand picked weights outperforms CASPERGeneral. CASPER earns a profit of $2.90 for every hand played, followed by CASPERGeneral with a profit of $2.20 for each hand. Random decisions resulted in exhausting the initial $1000 bankroll in only 30 hands, losing approximately -$30.80 for each hand played.

Both Pokibot and Simbot have also been tested against real opponents by playing online. Results reported by [5] indicate that Pokibot achieves a profit of +0.22 sb/h, i.e. a profit of $2.20 per hand, and Simbot achieves a profit of +0.19 sb/h or $1.90 profit per hand. These results are very similar to those obtained by CASPER.

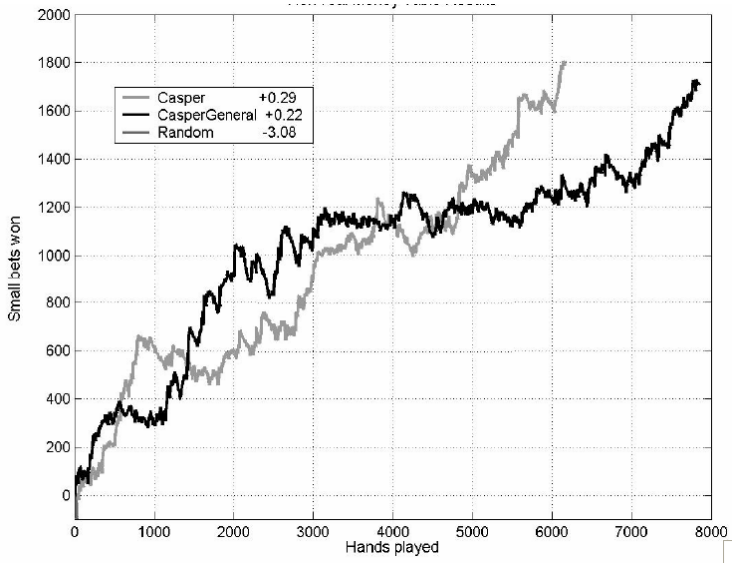

Fig. 1. CASPER vs. Real Opponents at the Online Play Money Tables

While we need to take caution in analysing the above results, it is safe to say that CASPER is consistently profitable at the 'play money' tables.

5.2 Real Opponents - Real Money

Because there is normally a substantial difference in the type of play at the 'play money' tables compared to the 'real money' tables it was decided to attempt to

get an idea of how CASPER would perform using real money against real opponents.

CASPER02 (with an increased case-base of 14,000 cases) with hand-picked feature weights that had achieved the best performance at the 'play money' tables was used to play at the 'real money' tables. The betting limit used was a small bet of $0.25 and a big bet of $0.50. CASPER started out with a bank roll of $100. The results are given in Figure 2.

Fig. 2. CASPER vs. Real Online Opponents at the Real Money Tables

CASPER achieves a small bet per hand value of -0.07. Therefore, CASPER now loses on average $0.02 per hand. The results indicate that while CASPER loses money very slowly it is now, nonetheless, unprofitable against these opponents. Due to the fact that real money was being used, fewer hands were able to be played and the experiment was stopped after CASPER had lost approx. $50. No results are published for Pokibot or Simbot challenging real opponents using real money.

6. Conclusion

In conclusion, CASPER, a case-based poker player has been developed that plays evenly against strong, adaptive poker-bots, plays profitably against non-adaptive poker-bots and against real opponents for play money. Two separate versions of CASPER were tested and the addition of extra cases to the case-base was shown to result in improvements in overall performance. It is interesting to note that

CASPER was initially unprofitable against the non-adaptive, aggressive poker-bots. One possible reason for this is that as CASPER was trained using data from players at the adaptive table it perhaps makes sense that they would play evenly, whereas players at the non-adaptive table tend to play much more loosely and aggressively. This means that while CASPER has extensive knowledge about the type of scenarios that often occur at the advanced table, this knowledge is weaker at the non-adaptive table as CASPER runs into unfamiliar situations.

Why then was CASPER not profitable on the real money tables. Two hypotheses could explain this. First, that people play poker very differently for real money than for play money. Since CASPER's case-base is derived from poker-bots that are playing for play money these cases are not representative of real money games. The average similarity of cases retrieved when playing against the bots is approx 90%, The average similarity of cases retrieved against real money players is approx. 50%. This is confirmation that CASPER is struggling to find good cases (*i.e.,* similar ones) to retrieve, therefore it's poor performance is not a surprise.

References

[1] Schaeffer, J., Culberson, J. Treloar, N., Knight, B., Lu, P. & Szafron, D. (1992). "A world championship caliber checkers program." Artificial Intelligence 53(2-3): 273-289.

[2] Schaeffer, J., Billings, D., Peña, L. & Szafron, D. (1999). "Learning to play strong poker." Proceedings of the ICML-99 Workshop on Machine Learning in Game Playing.

[3] Billings, D., Davidson, A., Schaeffer, J., & Szafron, D.. (2002). "The challenge of poker." Artificial Intelligence Vol(1-2): 201-240.

[4] Billings, D., Peña, L., Schaeffer, J. and Szafron, D. (1999). "Using probabilistic knowledge and simulation to play poker." Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence: 697-703.

[5] Davidson, A. (2002). Opponent modeling in poker: Learning and acting in a hostile and uncertain environment. Master's thesis, University of Alberta.

[6] Billings, D., Burch, N., Davidson A, Holte R, Schaeffer J, Schauenberg T. & Szafron, D. (2003). "Approximating game-theoretic optimal strategies for full-scale poker." IJCAI.

[7] Sklansky, D. (1994). "The Theory of Poker." Two Plus Two Publishing Las Vegas, NV, 1994

[8] Sklansky, D. & Malmuth, M. (1994). "Hold'em Poker for Advanced Players." Two Plus Two Publishing, Las Vegas, NV, 2nd ed.

[9] Korb, K. B., Nicholson A.E. & Jitnah, N. (1999). "Bayesian poker." UAI'99 - Proceedings of the 15th International Conference on Uncertainty in Artificial Intelligence, Sweden: 343-350.

[10] Dahl, F. A. (2001). A Reinforcement Learning Algorithm Applied to Simplified Two-Player Texas Hold'em Poker. Proceedings of the 12th European Conference on Machine Learning Springer-Verlag.

[11] Sandven, A. & Tessem, B. (2006). A Case-Based Learner for Poker. The Ninth Scandinavian Conference on Artificial Intelligence (SCAI 2006), Helsinki, Finland.

[12] Salim, M. & Rohwer, P. (2005). Poker Opponent Modeling. Indiana University: Personal communication.

[13] Rubin, J. & Watson, I. (2007). Investigating the Effectiveness of Applying Case-Based Reasoning to the game of Texas Hold'em. In, Proc. of the 20th. Florida Artificial Intelligence Research Society Conference (FLAIRS), Key West, Florida, May 2007. AAAI Press.