Investigating the Effectiveness of Applying Case-Based Reasoning to the game of Texas Hold'em

Jonathan Rubin[1] and Ian Watson[2]

Department of Computer Science, University of Auckland, New Zealand
[1]jrub001@ec.auckland.ac.nz
[2]ian@cs.auckland.ac.nz

Abstract

This paper investigates the use of the case-based reasoning methodology applied to the game of Texas hold'em. The development of a CASe-based Poker playER (CASPER) is discussed. CASPER uses knowledge of previous poker scenarios to inform a betting decision. 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 and profitably against other competition.

1. Introduction

The game of poker provides an interesting environment to investigate how to handle uncertain knowledge and issues such as dealing with chance and deception in a hostile environment. 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 which players must adhere to. Most games have precise goals and objectives which players must meet 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 which 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 (Schaeffer, Culberson et al. 1992).

Games such as poker on the other hand are classified as *stochastic*, *imperfect information* games. The game involves elements of chance (the actual cards which are dealt) and hidden information in the form of other player's *hole cards* (cards which only they can see). This ensures that players now need to make decisions with uncertain information present.

The focus of this paper is to investigate the application of CBR to the game of poker. We have developed a poker playing robot, nicknamed CASPER (CASe-based Poker playER), that attempts to use knowledge about 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.

The remainder of this paper is structured as follows, section two will detail related previous research, section three gives a brief introduction to the game of Texas hold'em. Sections four, five and six describe the design and implementation of CASPER. This is followed by the experimental results obtained (section seven) and a conclusion and discussion of future work in section eight.

2. Related Work

Over the last few years there has been a dramatic increase in the popularity of the game of Texas hold'em. This growing popularity has also sparked an interest in the AI community with increased attempts to construct poker robots (or bots), i.e. computerised poker players who play the game based on various algorithms or heuristics. Recent approaches to poker research can be classified into three broad categories:

Heuristic rule-based systems: which use various pieces of information, such as the cards a player holds and the amount of money being wagered, to inform a betting strategy.
Simulation/Enumeration-based approaches: which consist of playing out many scenarios from a certain point in the hand and obtaining the *expected value* of different decisions.
Game-theoretic solutions: which attempt to produce optimal strategies by constructing the game tree in which game states are represented as nodes and an agents possible decisions are represented as arcs.

* With special thanks to the University of Alberta Computer Poker Research Group.

The University of Alberta Computer Poker Research Group[1] are currently leading the way with poker related research, having investigated all of the above approaches. Perhaps the most well known outcome of their efforts are the poker bots nicknamed *Loki/Poki* (Schaeffer, Billings et al. 1999; Billings, Davidson et al. 2002).

Loki originally used expert defined rules to inform a betting decision. While expert defined rule-based systems can produce poker programs of reasonable quality (Billings, Davidson et al. 2002), 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 which 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 (Billings, Peña et al. 1999).

Loki was later rewritten and renamed *Poki* (Davidson 2002). A simulation-based betting strategy was developed which consisted of playing out many scenarios from a certain point in the hand and obtaining the *expected value* (EV) 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* was a consistent winner in the lower limit games and also performed well in the higher limit games where it faced tougher opposition (Billings, Davidson et al. 2002). More recently the use of game theory has been investigated in the construction of a poker playing bot. 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 (Billings, Burch et al. 2003).

There have also been numerous other contributions to poker research outside the University of Alberta Poker Research Group. Sklansky and Malmuth have detailed various heuristics for different stages of play in the game of Texas hold'em (Sklansky 1994; Sklansky and Malmuth 1994). 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, Nicholson et al. 1999) have produced a Bayesian Poker Program (BPP) which makes use of Bayesian networks to play five-card stud poker. (Dahl 2001) investigated the use of reinforcement learning for neural net-based agents playing a simplified version of Texas hold'em.

Finally, we have encountered relatively few attempts to apply the principles and techniques of CBR to the game of poker. (Sandven and Tessem 2006) constructed a case-based learner for Texas hold'em which they nicknamed Casey. Casey began with no poker knowledge and builds up a case-base for all hands that it plays. Sandven and Tessem report that Casey plays on a par with a simple rule-based system against three opponents, but loses when it faces more opponents. (Salim and Rohwer 2005) have attempted to apply CBR to the area of opponent modeling, i.e. trying to predict the hand strength of an opponent given how that opponent has been observed playing in the past. While CBR seems inherently suited to this particular type of task they report better performance by simply relying on long-term averages.

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 (Sklansky 1994).

The game of Texas hold'em is played in four stages, these include the *preflop*, *flop*, *turn* and the *river*. During each round all active players need to make a betting decision. Each betting decision is summarised below:

Fold: A player discards their hand and contributes no money to the pot. Once a player *folds* they are no longer involved in the current hand, but can still participate in any future hands.

Check/Call: A player contributes the least amount possible to stay in the hand. A *check* means that the player invests nothing, whereas a *call* means the player invests the least amount required greater than $0.

Bet/Raise: A player can invest their own money to the pot over and above what is needed to stay in the current round. If the player is able to *check,* but they decide to add money to the pot this is called a *bet*. If a player is able to *call*, but decides to add more money to the pot this is called a *raise*.

All betting is controlled by two imposed limits known as the small bet and the big bet. For example, in a $10/$20 game the small bet is $10 and all betting that occurs during the *preflop* and the *flop* are in increments of the small bet. During the *turn* and the *river* all betting is in increments of the big bet, $20. The number of bets made within each stage of the game is capped at a maximum of 5. All results detailed in this paper refer to a $10/$20 limit game[2]. Before the hand begins two forced bets are made, known as the *small blind* (half the small bet) and the *big blind* (one full small bet), to ensure that there is something in the

[1] http://www.cs.ualberta.ca/~games/poker/

[2] In no limit there is no restriction on the amount a player can bet.

pot to begin with. Each of the four game stages are summarised below:

Preflop: The game of Texas hold'em begins with each player being dealt two *hole cards* which only they can see. A round of betting occurs. Once a player has made their decision play continues in a clockwise fashion round the table. As long as there are at least two players left then play continues to the next stage. During any stage of the game if all players, except one, fold their hand then the player who did not fold their hand wins the pot and the hand is over.

Flop: Once the *preflop* betting has completed three community cards are dealt. Community cards are shared by all players at the table. Players use their hole cards along with the community cards to make their best hand. Another round of betting occurs.

Turn: The *turn* involves the drawing of one more community card. Once again players use any combination of their *hole cards* and the community cards to make their best hand. Another round of betting occurs and as long as there are at least two players left then play continues to the final stage.

River: During the *river* the final community card is dealt proceeded by a final round of betting. If at least two players are still active in the hand a *showdown* occurs in which all players reveal their *hole cards* and the player with the highest ranking hand wins the entire pot (in the event that more than one player holds the winning hand then the pot is split evenly between these players).

4. Casper System Overview

CASPER uses CBR to make a betting decision. This means that when it is CASPER's turn to act he evaluates the current state of the game and constructs a *target case* to represent this information. A target case is composed of a number of features. These features record 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 his case-base (i.e. his knowledge of past poker experiences) to try and find similar scenarios which may have been encountered. CASPER's case-base is made up of a collection of cases composed of their own feature values and the action which 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 the commercially available product Poker Academy Pro 2.5[3] 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 bot *Simbot*. Both *Pokibot* and *Simbot* are the result of an intensive knowledge engineering process. These poker bots have been used to generate the training data for CASPER. Approximately 7000 hands were played between various poker bots and each decision witnessed was recorded as a single case (or experience) in CASPER's case-base. Both bots have proven to be profitable against human competition in the past (Davidson 2002) so it is believed that the data obtained is of greater quality then it might be from other sources, such as free money games on the internet composed of real players. CASPER then reuses these recorded instances to make decisions at the poker table and therefore bypasses the intensive knowledge engineering effort required of other poker-bots.

[3] http://www.poker-academy.com/

5. Case Features

CASPER searches a different case-base for each separate 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 at a particular time are listed and explained in Table 1. The features listed were chosen by the authors because they are believed to capture important information needed to make a betting decision. These are the indexed features, which means that they are believed to be predictive of a case's outcome and by computing local similarity for each feature they are used to retrieve the most similar cases in the case-base. The first eight features are used in all case-bases, whereas the last four features are only used during the postflop stages. Each case is also composed of one outcome, which is the betting decision that was made.

The 'hand strength' feature, listed in table 1, differs somewhat for *preflop* and *postflop* stages of the game. During the *preflop* there exists 169 distinct card groups that a player could be dealt. These card groups were ordered from 1 to 169 based on their hand ranking, where 1 indicates pocket Aces (the best *preflop* hand) and 169 indicates a 2 and a 7 of different suits (the worst *preflop* hand). *Preflop* hand strength was then based on this ordering, whereas *postflop* hand strength refers to a calculation of immediate hand strength based on the *hole cards* a player has and the current *community cards* that are present. This value is calculated by enumerating all possible *hole cards* for a single opponent and recording how many of these hands are better, worse or equal to the current player's hand. *Positive potential* and *negative potential* are also calculated in this way with the addition that all possible future community cards are considered as well. For more details on hand strength and potential consult (Billings, Davidson et al. 2002; Davidson 2002).

6. Case Retrieval

Once a target case has been constructed CASPER needs to locate and retrieve the most similar cases it has stored in its case-base. The k-nearest neighbour algorithm is used to compute a similarity value for all cases in the case-base.

Each feature has a local similarity metric associated with it that evaluates how similar its value is to a separate case's value, where 1.0 indicates an exact match and 0.0 indicates entirely dissimilar.

Two separate similarity metrics are used depending on the type of feature. The first is the standard Euclidean distance function given by the following equation:

$$s_i = 1 - \left(|x_1 - x_2| \big/ MAX\_DIFF \right) \tag{1}$$

where x_1 refers to the target value, x_2 refers to the case value and *MAX_DIFF* is the greatest difference in values, given by the range in table 1.

The above Euclidean similarity metric produces smooth, continuous changes in similarity, however, for some features, minor differences in their values produce major changes in actual similarity, e.g. the 'Bets to call' feature. For this reason an exponential decay function, given by equation (2), has also been used for some features:

$$s_i = e^{-k\,(|x_1 - x_2|)}, \tag{2}$$

where, x_1 refers to the target value and x_2 refers to the source value and k is a constant that controls the rate of decay.

Feature:	Type:	Range:	Explanation:
Number of players	int	2 - 10	Number of active players at the beginning of the round (preflop, flop, turn or river).
Relative position	double	0.0 - 1.0	What order the player acts relative to other players at the table. 0.0 means the player is first to act in the round, 1.0 means the player is last to act.
Players in current hand	int	0 - 9	The number of players that have already acted and are still currently in the hand, i.e. players that have checked, bet, called or raised.
Players yet to act	int	0 - 9	The number of players that still need to make a future betting decision.
Bets committed	double	0.0 - 5.0	A multiple of the current bet size a certain player has committed to the pot. Small bets are used during the preflop and flop and big bets are used during the turn and river.
Bets to call	double	0.0 - 5.0	A multiple of the current bet size a certain player has to commit to the pot to stay in the hand. Small bets are used during the preflop and flop and big bets are used during the turn and river.
Pot Odds	double	0.0 - 0.5	The amount to call divided by the amount currently in the pot plus the amount needing to be called, a risk/reward measure.
Hand strength	double	0.0 - 1.0	A numerical measure of the strength of a player's hand. 0.0 represents the worst possible hand whereas 1.0 represents an unbeatable hand ("the nuts").
Positive potential[4]	double	0.0 - ~0.40	A numerical measure which represents the chance that a player who does not currently hold the best hand will improve to the best hand after future community cards are dealt.
Negative potential[4]	double	0.0 - ~0.30	A numerical measure which represents the chance that a player currently holding the best hand no longer holds the best hand after future community cards are dealt.
Small bets in pot	double	0.0 - ~300.0	The total amount in the pot divided by the value of the small bet size.
Previous round bets	int	0 - 5	How many bets or raises occurred during the previous betting round.
Action	char	{f, k, c, b, r}	A character representing the decision which was made. f = fold, k = check, c = call, b = bet, r = raise.

Table 1. *Preflop* and *postflop* case features.

[4] Not used during the *river* as there are no further betting rounds.

Global similarity is computed as a weighted linear combination of local similarity, where higher weights are given to features that refer to a player's hand strength as well as positive and negative potential. All weights were hand picked by the authors and fell in the range of 0 – 100. A default value of 5 was used for most features, while features we felt were more salient, such as 'hand strength' and 'positive and negative potential', were assigned values in the approximate range of 50 - 80. The following equation is used to compute the final similarity value for each case in the case-base:

$$\sum_{i=1}^{n} w_i x_i \Big/ \sum_{i=1}^{n} w_i \ , \qquad (3)$$

where x_i refers to the ith local similarity metric in the range 0.0 to 1.0 and w_i is its associated weight, in the range 0 – 100.

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 which exceed a similarity threshold of 97% are recorded. Each action is summed and then divided by the total number of similar cases which results in the construction of a probability triple *(f, c, r)* which gives the probability of folding, checking/calling or betting/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 his *hole cards* and sees **A♥-A♠**. After a search of his *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 which was generated.

7. Results

CASPER was evaluated by playing other poker bots provided through the commercial software product Poker Academy Pro 2.5. CASPER was tested 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 his own he was also tested against non-adaptive, but loose/aggressive opponents. A loose opponent is one that plays a lot more hands, whereas aggressive means that they tend to bet and raise more than other players. All games were \$10/\$20 limit games which consisted of 9 players. All players began with a starting bankroll of \$100,000.

The adaptive table consisted of different versions of the University of Alberta's poker bots: Pokibot and Simbot. Figure 1 records the amount of small bets won at the adaptive table over a period of approximately 20,000 hands. Two separate versions of CASPER were tested separately against the same competition. Casper02 improves upon Casper01 by using a larger case-base, generated from approximately 13,000 poker hands. A poker bot that makes totally random betting decisions was also tested separately against the same opponents as a baseline comparison.

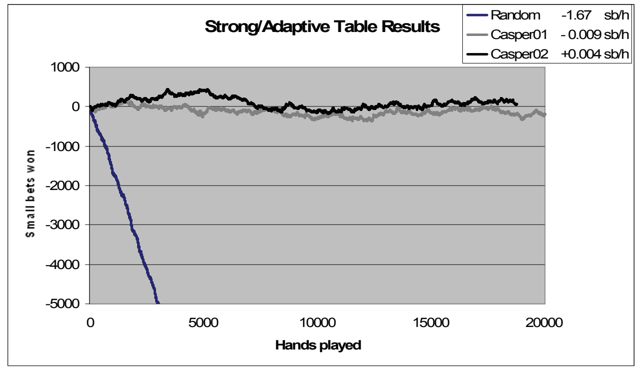

Figure 1: Casper's results at the adaptive table.

While Casper01 concludes with a slight loss and Casper02 concludes with a slight profit, figure 1 suggests that both versions approximately break even against strong competition, whereas the random player exhausted its bankroll of \$100,000 after approximately 6,000 hands[5]. Casper01's small bets per hand (sb/h) value is -0.009 which indicates that Casper01 loses about \$0.09 with every hand played. Casper02 slightly improves upon this by winning approximately \$0.04 for every hand played. Random play against the same opponents produces a loss of \$16.70 for every hand played.

The second table consisted of different versions of Jagbot, a non-adaptive, loose/aggressive rule-based player. Figure 2 records the amount of small bets won over a period of approximately 20,000 hands. Once again a bot which makes random decisions was also tested separately against the same competition as a baseline comparison for CASPER.

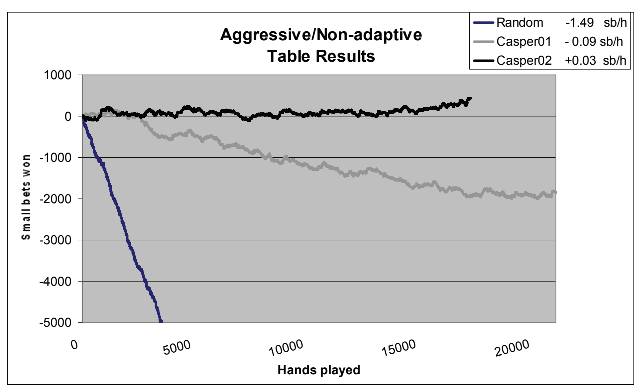

Figure 2: Casper's results at the non-adaptive table.

Figure 2 indicates that the first version of Casper is unprofitable against the non-adaptive players, losing

[5] Not all data points are shown to improve clarity.

approximately \$0.90 for each hand played. Casper02 shows a considerable improvement in performance compared to Casper01. With more cases added to the case-base, Casper02 produces a profit of +0.03 sb/h, or \$0.30 for each hand played. Once again the random player exhausted its initial bankroll after approximately 7000 hands, losing on average \$14.90 for each hand played.

8. Conclusions and Future Work

In conclusion, CASPER, a case-based poker player has been developed that plays evenly against strong, adaptive competition and plays profitably against non-adaptive opponents. The results suggest that it is possible to record instances of games from strong players and reuse these to obtain similar performance without the need for an intensive knowledge engineering effort.

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 opposition. 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 situations which he is not familiar with. For example, for a random sample of 10,000 hands there were 86 occurrences where Casper01 didn't exceed the similarity threshold during the *preflop* stage at the adaptive table, whereas this figure increases to 222 occurrences at the non-adaptive table. CASPER must therefore make a betting decision at the non-adaptive table using less similar scenarios. With the addition of extra cases these values drop to 38 and 45 for the *preflop* stage at the adaptive and non-adaptive table respectively.

While CASPER does achieve slight profits against various opposition, it appears as though there are still many improvements necessary to ensure that CASPER improves its profitability. Some areas of possible future work are:

- Investigating optimal feature weightings using a genetic algorithm and self play experiments.
- Further investigation of CASPER's level of play based on the size of its case-base.
- Improving the case-representation by considering the actual outcome that occurred rather than simply the action that was taken.
- Testing CASPER against real opponents.
- Adding opponent modeling capabilities.

References

Billings, D., N. Burch, et al. 2003. *Approximating game-theoretic optimal strategies for full-scale poker*. Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence.

Billings, D., A. Davidson, et al. 2002. *The challenge of poker*. Artificial Intelligence **134**(1-2): 201-240.

Billings, D., L. Peña, et al. 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.

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.

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

Korb, K. B., A. E. Nicholson, et al. 1999. *Bayesian poker*. UAI'99 - Proceedings of the 15th International Conference on Uncertainty in Artificial Intelligence, Sweden: 343-350.

Salim, M. and P. Rohwer 2005. *Poker Opponent Modeling*. Indiana University**:** Personal communication.

Riesbeck, C. and R. Schank 1989. *Inside Case-Based Reasoning*. Hillsdale, NJ.: Lawrence Erlbaum.

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

Schaeffer, J., D. Billings, et al. 1999. *Learning to play strong poker*. Proceedings of the ICML-99 Workshop on Machine Learning in Game Playing.

Schaeffer, J., J. Culberson, et al. 1992. *A world championship caliber checkers program*. Artificial Intelligence **53**(2-3): 273-289.

Sklansky, D. 1994. *The Theory of Poker*. Las Vegas, NV.: Two Plus Two Publishing.

Sklansky, D. and M. Malmuth 1994. *Hold'em Poker for Advanced Players*. Las Vegas, NV.: Two Plus Two Publishing.

Watson, I. 1997. *Applying case-based reasoning: techniques for enterprise systems*. San Francisco, CA.: Morgan Kaufmann Publishers Inc.