

Poker as a Testbed for Machine Intelligence Research

Darse Billings, Denis Papp, Jonathan Schaeffer, Duane Szafron

{darse, dpapp, jonathan, duane}@cs.ualberta.ca

Department of Computing Science

University of Alberta

Edmonton, Alberta

Canada T6G 2H1

ABSTRACT

For years, games researchers have used chess, checkers and other board games as a testbed for machine intelligence research. The success of world-championship-caliber programs for these games has resulted in a number of interesting games being overlooked. Specifically, we show that poker can serve as a better testbed for machine intelligence research related to decision making problems. Poker is a game of imperfect knowledge, where multiple competing agents must deal with risk management, agent modeling, unreliable information and deception, much like decision-making applications in the real world. The heuristic search and evaluation methods successfully employed in chess are not helpful here. This paper outlines the difficulty of playing strong poker, and describes our first steps towards building a world-class poker-playing program.

Keywords: poker, imperfect information, opponent modeling, computer games

1. Introduction

Why study computer games? By writing programs that play games, some insights can be gained about machine intelligence. These lessons can then be used to develop useful non-game programs. Researchers have spent a lot of time and effort on board games such as chess and checkers. These games all share the property that high performance can be achieved by brute-force search. This emphasis on search was taken to the extreme by the *Deep Blue* chess machine, which analyzed 200 million positions per second in its May 1997 match against World Chess Champion Garry Kasparov. This achievement only confirmed the effectiveness of brute-force search for some application domains. However, this result has been anticipated for several decades. Can computer games give us any fresh insights into machine intelligence, beyond brute-force search?

We believe that the answer to this question is yes. However, real progress can only be made if we study games in which search is not the major criteria for success. Instead, we need to mimic real-world applications that are perceived to require intelligent behaviour. Activities such as financial trading, business negotiations, and forecasting (from weather to politics) meet this criteria. The first column of Table 1 shows some of the characteristics of these applications from the AI point of view. We are not claiming that these are the only activities of interest, just that they are important considerations for a wide range of interesting problem domains. Unfortunately, games like chess and checkers do not have these characteristics, or involve them only in limited ways. Can these activities be studied in the context of

computer games, and if so, what games?

General Application Problem	Problem Realization in Poker
imperfect knowledge	opponents' hands are hidden
multiple competing agents	many competing players
risk management	betting strategies and their consequences
agent modeling	identifying patterns in opponent's play and exploiting them
deception	bluffing and varying style of play
unreliable information	taking into account your opponents' deceptive plays

Table 1. Characteristics of AI problems and how they are exhibited by poker.

We are currently studying the game of poker and are attempting to build a high-performance poker program that is capable of beating the best human players. As shown in the second column of Table 1, poker exhibits all of the activities we are interested in studying on at least some level.

Certain aspects of poker have been extensively studied by mathematicians and economists but, surprisingly, very little work has been done by computing scientists. There are two main approaches to poker research. One approach is to use simplified artificial variants [vNM44] or simplified real variants [Ank81, SS92] that are easier to analyze. For example, one could use only two players or constrain the betting rules. The other approach is to pick a real variant, but to combine mathematical analysis, simulation and ad-hoc expert experience. Expert players with a penchant for mathematics are usually involved in this approach (for example, [SM94]).

Simplification is a common technique for solving difficult problems. However, we must be careful that simplification does not remove the complex activities that we are interested in studying. For example, Findler worked on and off for 20 years on a poker-playing program for 5-card draw poker [Fin77]. His approach was to model human cognitive processes and build a program that could learn. Unfortunately his simplified approach nullified many of the potential benefits of his research [Bi195].

Recently, Koller and Pfeffer have been investigating poker [KoP97] from a theoretical point of view. They implement the first practical algorithm for finding optimal randomized strategies in two-player imperfect information competitive games. This is done in their *Gala* system, a tool for specifying and solving problems of imperfect information. Their system builds trees to find the game-theoretic optimal (but not maximal) strategy, however only vastly simplified versions of poker can be solved due to the size of trees being built. The authors state that "...we are nowhere close to being able to solve huge games such as full-scale poker, and it is unlikely that we will ever be able to do so."

We have chosen to study the game of Texas Hold'em, the poker variation used in the annual World Series of Poker Championships. It is considered to be the most strategically difficult poker variant that is widely played, and requires all of the complex activities listed in Table 1. For example, the best risk management strategy in the world cannot compensate for a lack of deception, since human opponents are quick to exploit predictable players, no matter how strong they might otherwise be. Our objective is to build a program which handles all aspects of poker well enough to play at world-championship caliber. If we are successful, then the insights we gain should have wide applicability to real applications that require similar activities.

This paper describes our first steps towards building a strong poker program, called *Lokibot*. Section 2 gives the rules of Texas Hold'em. Section 3 discusses the requirements of a strong Hold'em program and provides evidence that all of the activities listed in Table 1 are necessary to play strong poker. Section 4

describes the *Lokibot* program and Section 5 gives some initial performance assessments. Section 6 discusses ongoing work on this project.

The research contributions of this paper include:

- showing that poker can be a testbed of real-world decision making,
- identifying the major requirements of high-performance poker,
- presenting new enumeration techniques for hand-strength and potential, and
- demonstrating a working program that successfully plays "real" poker.

2. Texas Hold'em

A hand of Texas Hold'em begins with the *pre-flop*, where each player is dealt two *hole* cards, face down, followed by the first round of betting. Then three community cards are dealt face up on the table, called the *flop*, and the second round of betting occurs. On the *turn*, a fourth community card is dealt face up and another round of betting ensues. Finally, on the *river*, a fifth community card is dealt face up and the fourth (final) round of betting occurs. All players still in the game turn over their two hidden cards for the *showdown*. The best five card poker hand formed from the two hole cards and the five community cards wins the pot. If a tie occurs, the pot is split. Typically Texas Hold'em is played with 8 to 10 players.

Limit Texas Hold'em uses a structured betting system, where the order and amount of betting is strictly controlled on each betting round. There are two denominations of bets, called the small bet and the big bet. For simplicity, we will use a value of \$10 for the small bet and \$20 for the big bet. In the first and second betting rounds (pre-flop and flop), all bets and raises are \$10, while in rounds three and four (turn and river), they are \$20. In general, when it is a player's turn to bet, one of five options is available: *fold* (withdraw from the hand, leaving all previously wagered money in the *pot*), *call* (match the current outstanding bet; if there is no current bet, one is said to *check*), or *raise* one bet (put the current bet plus one into the pot; if there is no current bet, one is said to *bet*). There is usually a maximum of three raises allowed per betting round. The betting option rotates clockwise until each player that has not folded has put the same amount of money into the pot for the current round, or until there is only one player remaining. In the latter case, this player is the winner and is awarded the pot without having to reveal their cards.

There is a strategic advantage to being the last bettor in any given round, so to maintain fairness, the order of betting is rotated clockwise after each hand.

3. Requirements for a World-Class Poker Player

We have identified several key components (modules) that incorporate some of the required activities of a strong poker player and address most of the six characteristics listed in Table 1. However, these components are not independent. They must be continually refined as new activities are supported.

Hand strength: assesses how strong your hand is in relation to what other players may hold. Hand strength is computed on the flop, turn and river. At a minimum, hand strength is a function of your cards and the community cards that have been dealt. A better hand strength computation takes into account the number of players still in the game, position at the table, and history of betting in the hand. An even better model considers different probabilities for each hidden hand, based on the relative chance of each hand being played to the current point in the game. This model may be improved by varying the hidden hand probabilities for each player depending on the opponent's model of play for that player.

Hand potential: assesses the probability of the hand improving (or being overtaken) as additional community cards appear. For example, having four cards in the same suit does not count toward hand strength, but has good potential to become a winning flush as more community cards are dealt.

At a minimum, hand potential is a function of your cards and the community cards that have already been dealt. However, a better model can be evolved as capabilities are added to the program, similar to the hand strength computation described above.

Betting strategy: determines whether to fold, call/check, or bet/raise in any given situation. A minimum model is based on hand strength. Refinements consider hand potential, pot odds, bluffing, opponent modeling and unpredictability. *Pot odds* is an important concept that differentiates poker from many other games and contributes to its usefulness as a testbed for concepts in the real world. Pot odds is the comparison of your winning chances to the expected return from the pot. For example, if there is only a 20% chance that we have the best hand on the river, should we fold, call or bet? The correct answer is that we have not given enough information to answer the question. Assume the pot contains \$100 after the only opponent bets \$20. If you call in this situation, you will lose 4 times out of 5, for an additional cost of \$80. However, you will win 1 time out of 5 for a profit of \$100. Therefore, under these assumptions, you should call, resulting in an average profit of \$4 per hand. However, if the pot only contained \$60, you should fold, since calling would yield an average loss of \$4 per hand. Notice that an accurate computation of your winning chances is necessary. Such a computation requires a sophisticated assessment of hand strength, as described above. Even with an accurate hand strength computation, the game theoretic optimal folding/calling strategy may not be the best decision in practice, where bluffing, opponent modeling and unpredictability may be used to improve your betting strategy.

Bluffing: allows you to make a profit from weak hands. Even if you only break even on the bluffing plays, the false impression created about your play may improve the profitability of subsequent hands. Thus, bluffing is critical to successful play. Game theory can be used to compute a theoretical optimal bluffing frequency in certain situations. The minimal bluffing system merely bluffs this percentage of hands. In practice, you need to be able to predict the probability that your opponent will call in order to identify profitable bluffing opportunities. The better your opponent models are, the better your bluffing strategy will be.

Opponent modeling: allows you to determine a likely probability distribution for your opponent's hidden cards or betting strategy. A minimal opponent model might use a single model for all opponents in a given hand. Before the flop, a weighting system may be used to estimate the probability of possible holdings for all players who do not fold. After the flop, a second set of probabilities may be used for all opponents who do not fold, based on the three community cards that have been dealt. Opponent modeling may be improved by modifying the probabilities based on a classification of each opponent (e.g. weak/strong, passive/aggressive), betting history, and collected statistics. Opponent modeling has been attempted in two-player games but with limited success [CM95]. In poker, however, it is essential to success.

Unpredictability: makes it difficult for opponents to form an accurate model of your strategy. By varying playing strategy over time (e.g. pre-flop hand selection, variable bluffing rate), opponents may be induced to make mistakes based on an incorrect model.

In addition, there is a number of less immediate concerns which may not be necessary to play reasonably strong poker, but may be required for world-class play.

This paper focuses on the issues of hand strength, hand potential and betting strategy. Other issues are the subject of on-going research.

4. *Lokibot*

Lokibot handles its play differently at the pre-flop, flop, turn and river. The play is controlled by two components: an evaluation of the hand and a betting strategy. The strategy is influenced both by the pot odds and our model of the opponent.

4.1. Pre-flop Evaluation

The hand strength for pre-flop play has been extensively studied in the poker literature (for example, [SM94]). These works attempt to explain the play in human understandable terms, by classifying all the initial two-card pre-flop combinations into nine betting categories. For each hand category, a suggested betting strategy is given, based on the strength of the hand, the number of players in the game, the position at the table, and the type of opponents. For a poker program, these ideas could be implemented as an expert system, but a more general approach would be preferable.

For the initial two cards, there are $\{52 \text{ choose } 2\} = 1326$ possible combinations, but only 169 distinct hand types. For each one of the 169 possible hand types, a simulation of 1,000,000 poker games was done against nine random opponents. This produced a statistical measure of the approximate *income rate* for each starting hand. A pair of aces had the highest income rate; a 2 and 7 (of different suits) had the lowest income rate for a 10-player simulation. There is a strong correlation between our simulation results and the pre-flop card ordering given in [SM94] (although there are a few interesting differences).

4.2. Hand Evaluation

Critical to the program's performance on the flop, turn and river is an assessment of the current strength of the program's hand. Enumeration techniques can provide an accurate estimate of the probability of currently holding the strongest hand.

For example, suppose our starting hand is $A\spadesuit-Q\clubsuit$ and the flop is $3\heartsuit-4\clubsuit-J\heartsuit$. There are 47 remaining unknown cards and there are $\{47 \text{ choose } 2\} = 1,081$ possible hands an opponent might hold. To estimate hand strength, the enumeration technique gives a percentile ranking of our hand. We simply count the number of possible hands that are better than ours (any pair, two pair, A-K, or three of a kind: 444 hands), how many hands are equal to ours (9 possible remaining A-Q combinations), and how many hands are worse than ours (628). Counting ties as half, this corresponds to a percentile ranking, or hand strength (HS), of 0.585. In other words there is a 58.5% chance that our hand is better than a random hand. This measure is with respect to one opponent but can be extrapolated to multiple opponents by raising it to the power of the number of active opponents. Against five opponents with random hands, the adjusted hand strength (HS5) is $.5855 = .069$. Hence, the presence of additional opponents has reduced the likelihood of our having the best hand to only 6.9%.

In practice, hand strength alone is insufficient to assess the quality of a hand. Consider the hand $8\spadesuit-7\heartsuit$ with a flop of $9\spadesuit-6\clubsuit-2\spadesuit$. The probability of having the strongest hand is very low, even against one random opponent. On the other hand, there is tremendous potential for improvement. With two cards yet to come, any \spadesuit , 10, or 5 will give us a straight or a flush. Hence there is a high probability that this hand will improve substantially in strength, so the hand has a lot of value. We need to be aware of the potential changes of hand strength.

In addition to this positive potential (Ppot) of pulling ahead when we are behind, enumeration can also compute the negative potential (Npot) of falling behind if we are ahead. For each of the possible 1,081 opposing hands, we consider the $\{45 \text{ choose } 2\} = 990$ combinations of the next two cards. For each subcase we count how many combinations of upcoming cards result in us being ahead, behind or tied.

The potential for $A\spadesuit-Q\clubsuit / 3\heartsuit-4\clubsuit-J\heartsuit$ is shown in Table 2. The table shows, for cases where we were ahead, tied or behind after five cards, what the result would be after seven cards. For example, if we did not have the best hand after five cards, then there are 91,981 combinations of cards (pre-flop and two cards to come) for the opponents that will give us the best hand. Of the remaining hands, 1,036 will leave us tied with the best hand, and 346,543 will leave us behind. In other words, if we are behind we have roughly a 21% chance of winning against one opponent.

cards				
	Ahead	Tied	Behind	Sum
Ahead	449,005	3,211	169,504	621,720 = 628x990
Tied	0	8,370	540	8,910 = 9x990
Behind	91,981	1,036	346,543	439,560 = 444x990
Sum	540,986	12,617	516,587	1,070,190 = 1,081x990

Table 2. A♦-Q♣ / 3♥-4♠-J♥ potential.

We use these values to generate Ppot and Npot. If $T_{\{row,col\}}$ refers to the values in the table (for brevity we use B, T, A, and S for Behind, Tied, Ahead, and Sum) then Ppot and Npot are calculated by:

$$Ppot = (T_{\{B,A\}} + T_{\{B,T\}}/2 + T_{\{T,A\}}/2) / (T_{\{B,S\}} + T_{\{T,S\}}/2)$$

$$Npot = (T_{\{A,B\}} + T_{\{A,T\}}/2 + T_{\{T,B\}}/2) / (T_{\{A,S\}} + T_{\{T,S\}}/2)$$

In the example Ppot is .208 and Npot is .274. The calculation for one card lookahead is exactly the same as the above calculation, except there are only 45 possible upcoming cards instead of 990 (or 44 if we are on the turn). With only one card to come on the turn, Ppot is .108 and Npot is .145.

By enumerating all possible card combinations, the program uses a brute-force approach to calculating hand strength and potential. The calculations are easily done in real-time and provide accurate probabilities that take into account every possible scenario. Hence the calculation gives smooth and robust results.

4.3 Weighting the Enumeration

So far our calculations assume that all opponent hands are equally likely. In reality, this is not the case. Many weak hands like 4♥-J♣ would have been folded before the flop. However, with the example flop of 3♥-4♠-J♥, these hidden cards make a strong hand that skews the hand evaluations.

Accuracy of the estimates also depend strongly on models of our opponents. Ultimately, we want a different set of weights for *each* possible starting hand for each opponent. These weights could then be adjusted depending on the opponent's playing style. For example, raising on the flop probably indicates a strong hand that should be reflected in the weightings. We would then apply the appropriate weight for each of the 1,081 possible subcases when calculating hand strength and potential.

Although *Lokibot* treats all opponents the same, it was designed to support generalized opponent modeling. Currently the common weights are based on the simulations of the 169 different starting hand types, providing a reasonable starting template for unknown players.

4.4 Betting Strategy

Hand strength and potential are combined into *effective hand strength* (EHS):

$$EHS = HS_n + (1 - HS_n) \times Ppot$$

where HS_n is the adjusted hand strength for n opponents and Ppot is the positive potential. This formula means that EHS is the probability that we are ahead, and in those cases where we are behind there is a Ppot chance that we will pull ahead. Currently, EHS is compared to some thresholds to determine when to bet. For example, with an EHS greater than 0.5 we can say there is a reasonable chance we are ahead of our opponents and will bet if no other opponent has bet. This is an optimistic estimate because we only consider positive potential. Npot is not considered for two reasons. First, we do not know if our opponent

will play. Second, in many situations where we calculate a high N_{pot} , it is often a better strategy to bet/raise to scare the opponent out of the hand.

Determining if the pot is large enough and whether we have enough equity to warrant calling a bet is different than deciding when to bet. This decision is made by comparing P_{pot} against the pot odds, where

$$pot_odds = bets_to_us / (bets_in_pot + bets_to_us) .$$

We call when $P_{pot} > pot_odds$. Note that even on the flop we use only one card look ahead for P_{pot} . If we examine the situation two cards in the future we must consider whether we will face another bet (or more) after the first card. Pot_odds is based on the immediate situation. In the original example, if there are five opponents and we are first to act, EHS is 0.15 so we check. If the first opponent behind us bets \$10, two others call, and the fourth raises \$10, then it is \$20 to us and the pot is \$175. Therefore pot_odds is $20 / (175+20) = 0.103$, so we call (P_{pot} is 0.108).

5. Experiments

A variety of different experimental methods have been used to measure the development of *Lokibot*. These include self-play simulations, play against human opposition, and play against other computer programs. Each type of evaluation has limitations, reinforcing the need for a wide range of experiments and testing methods.

Self-play simulations offer a convenient method for the comparison of two or more versions of the program. In addition to verifying that a certain enhancement has a beneficial effect, it is possible to quantify the contribution made by each new component to the system. Since all participants in the simulated game are versions of the program, play can proceed at a rapid pace, and results can be based on large, statistically significant, sample sizes. Moreover, these closed experiments can be used as a vehicle for exploring the interdependencies of program features. A combination of competing factors can produce different results than might be expected from looking at each variable in isolation. Exploring these results can help identify weaknesses in the current system and suggest areas to focus on, providing some direction for future work.

The self-play simulations use a duplicate tournament system, based on the same principle as duplicate bridge. Since each hand can be played with no memory of preceding hands, it is possible to replay the same deal, but with the participants holding a different set of hole cards. Our tournament system simulates a ten-player game, where each deal is replayed ten times, shuffling the seating arrangement each time so that every participant has the opportunity to play each set of hole cards once. This arrangement greatly reduces the "luck element" of the game, since each player will have the same number of good and bad hands. The differences in the performance of players will therefore be based more strongly on the quality of the decisions made in each situation. This large reduction in natural variance means that meaningful results can be obtained with a much smaller number of trials than a typical game setting.

One simple application of a self-play simulation would be to play five copies of a new version against five copies of an older version, differing only in the addition of one new feature. If the new component has improved the program, then the newer version will win against the older version. The average margin of victory, in terms of expected number of bets per hand, can also give a preliminary indication of the relative value of the new enhancement.

However, there are limitations to how much can be concluded from a single experiment, since it is representative of only one particular type of game and style of opponent. It is quite possible that the same feature will perform much worse (or much better) in a game against human opposition, for example. A wider variety of testing is necessary to get an accurate assessment of the new feature. One approach is to change the context of the simulated game. The next self-play experiment might include a number of players who employ a different style of play, such as a more liberal selection of starting hands. If the new feature is successful over a wide variety of game types, we will have a more reliable indication of the

value of that concept, with a metric to quantify its contribution.

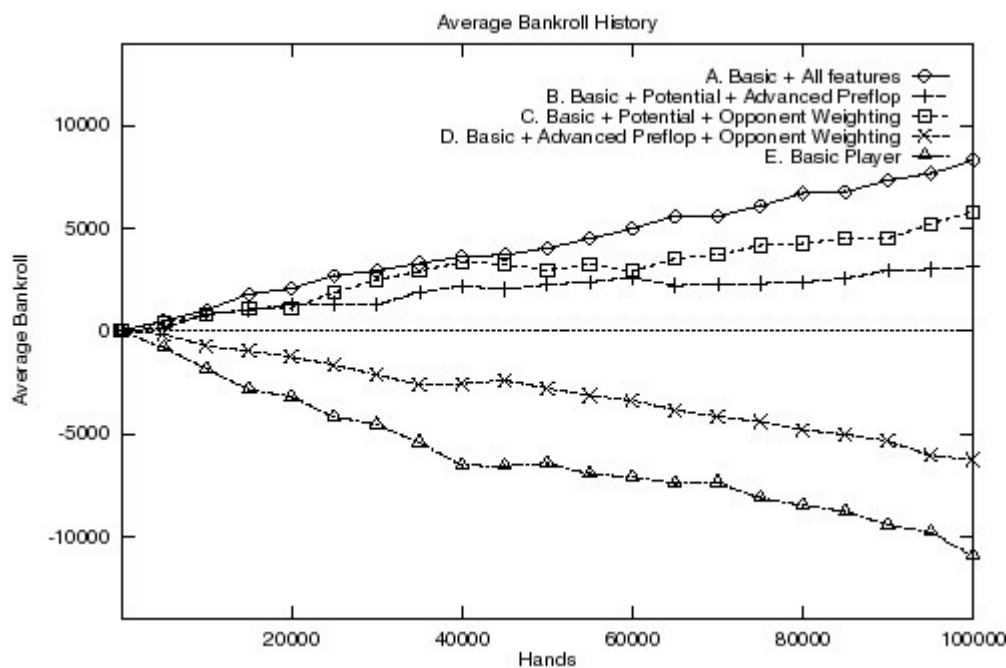


Figure 1. Experiments with different versions of *Lokibot*.

As an example of a self-play experiment, Figure 1 shows the results of a tournament with five different versions of *Lokibot*. The average bankroll size (profit) is plotted against the number of hands played. Player A is the most advanced version of the program, including three major components that the most basic player does not have. Player E is a basic player, having no advanced features. The other three versions are the same as Player A, but with one of the major components removed. Player B lacks an appropriate weighting of subcases, using a uniform distribution for all possible opponent hands. Player C uses a simplistic pre-flop hand selection method, rather than the advanced system which accounts for player position and number of opponents. Player D lacks the computation of hand potential, which is used in modifying the effective hand strength and calling with proper pot odds.

As expected, the complete system performs the best, while the basic system loses the most. The best program earned approximately +0.08 small bets per hand, while the worst lost at a rate of -0.11 small bets per hand.

Within the context of this particular experiment, the use of hand potential had the greatest impact on the strength of the program, since Player D, which lacked that component, performed poorly. Player B, missing the appropriate weighting of subcases, was still able to win against this field of opponents, but did not perform nearly as well as the version having this feature. Player C, differing only in the use of the advanced pre-flop hand selection method, did not lose much compared to the other weakened versions.

It is important not to over-interpret the results of a single experiment. In this particular tournament, all of the participants are computer players with fairly conservative styles. It is quite possible that the consequences of each change would be different against a field of opponents who employ different playing styles. For example, against several human players, the weighting function may have a much bigger impact than the use of hand potential.

Lokibot must also be tested in more realistic games against human opposition. For this purpose, the program participates in an on-line poker game, running on IRC (Internet Relay Chat). Human players connect to the server and participate in games. No real money is at stake, but statistics on each player are maintained. Certain games are reserved for players who have earned enough virtual dollars to qualify, and those games are usually taken more seriously than the games open to all players. This provides an

environment with several games, differing in styles of play and skill level.

Early versions of *Lokibot* had mixed results on the IRC server, but played at about the same level as the average human participant in the open games, roughly breaking even over the course of about 12,000 hands. When it qualified for the stronger game it lost slowly, averaging about -0.05 small bets per hand dealt, based on roughly 2,000 hands. This is not a large enough sample size for conclusive results, but strongly suggests it was a losing player overall in these games.

The most recent versions of *Lokibot* have performed much better in the open games, averaging about +0.20 small bets per hand over 3,500 hands dealt, which is comparable to a solid human player (probably ranking in the top 10% of IRC players).

A third form of competition was introduced against other computer programs on the IRC server. Four programs participated, using three copies of each in a 12-player game. Two programs, *R00lbot* and *Lokibot*, were clearly dominant over the other two, *Xbot* and *Replicat*, with the more established *R00lbot* winning overall. Over 10,000 hands, *Lokibot* averaged about +0.03 small bets per hand. It should be noted, however, that this competition is representative of only one type of game, where all the players are quite conservative. *Replicat* in particular performed much better in the open games against human opposition than in this closed experiment.

A final important method of evaluation is the critique of expert human players. Experts can review the play of the computer and determine if certain decisions are "reasonable" under the circumstances, or are indicative of a serious weakness or misconception. Based on this opinion, it appears to be feasible to write a program that is stronger than the average human player in a casino game, although *Lokibot* has not yet achieved that level. Whether it will be possible to design a system capable of world-champion-caliber play remains an open question.

6. Work in Progress

Lokibot still suffers from some obvious problems. Most importantly, it is a predictable player that reacts the same in a given situation irrespective of any historical information. This leaves it open to exploitation by an opponent who has deduced its simplistic playing style. The two major areas requiring improvement are opponent modeling and betting strategy. Both of these topics are open-ended, and will provide interesting challenges for future work.

The most important foreseeable advance is opponent modeling. When *Lokibot* is better able to infer likely holdings for the opponent, it will be capable of much better decisions. The hand strength and potential calculations will use a different table of weights for each particular opponent. Now the specific actions of that opponent can be taken into consideration, as well as historical and statistical information gathered on this opponent from previous games. A wide variety of properties can be measured and applied, such as betting frequencies, known bluffs, recent trends, etc. Additionally for each opponent we will also compute statistics to measure the betting strategy and thresholds for the various betting actions.

Betting strategy is similarly a very broad concept. The current system is simplistic and predictable (it will always act the same in a given situation). A significantly better betting system would bluff with high potential hands and occasionally bet a strong hand weakly. It would also predict opponent responses in order to choose the best practical action. Unpredictability and other advanced betting strategies can be incorporated.

The infrastructure is in place to incorporate these features. *Lokibot* is changing on a daily basis. It is only six months old and already at a level that exceeds our initial expectations. We understand many of the weaknesses in the program, but do not yet know if all of them can be addressed sufficiently to produce a world-class poker player.

References

Ank81	N. Ankeny, <i>Poker Strategy: Winning with Game Theory</i> , Basic Books, Inc., 1981.
Bil95	D. Billings, <i>Computer Poker</i> , M.Sc. thesis, Dept. of Computing Science, University of Alberta, 1995.
CM95	D. Carmel and S. Markovitch, Incorporating Opponent Models into Adversary Search, <i>AAAI</i> , 1995, 120-125.
Fin77	N. Findler, Studies in Machine Cognition Using the Game of Poker, <i>Communications of the ACM</i> 20, 4 (1977), 230-245.
KoP97	D. Koller and A. Pfeffer, Representations and Solutions for Game-Theoretic Problems, <i>Artificial Intelligence</i> 94, 1-2(1997), 167-215.
SM94	D. Sklansky and M. Malmuth, <i>Hold'em Poker for Advanced Players</i> , Two Plus Two Publishing, 1994.
SS92	M. Sakaguchi and S. Sakai, Solutions of Some Three-person Stud and Draw Poker, <i>Mathematics Japonica</i> 37, 6(1992), 1147-1160.
vNM44	von Neumann and O. Morgenstern, <i>Theory of Games and Economic Behavior</i> , Princeton University Press, 1944.