

USING BAYESIAN DECISION NETWORKS TO PLAY TEXAS HOLD'EM POKER

Ann E. Nicholson¹ and Kevin B. Korb² and Darren Boulton

Faculty of Information Technology, Monash University, Australia

ABSTRACT

Poker is an ideal vehicle for testing automated reasoning under uncertainty. It introduces uncertainty through physical randomization by shuffling and through incomplete information about opponents' hands. Another source of uncertainty is the limited knowledge of opponents, their betting strategies, tendencies to bluff, play conservatively, reveal weaknesses, etc. Furthermore, poker is well known as a game of psychology, with success coming from a combination of mathematical accuracy and effective prediction of one's opponent. All of these uncertainties must be assessed accurately and combined effectively for any reasonable level of skill in the game to be achieved, since good decision making is highly sensitive to those tasks. We describe our Bayesian Poker Program (BPP), which uses a Bayesian decision network to model the program's poker hand, the opponent's hand and the opponent's playing behaviour conditioned upon the hand, and betting curves to randomise betting actions. BPP has been developed incrementally for 5-card stud poker since 1993. Here we describe its adaptation to play Texas Hold'em and a variety of advances which have improved BPP's play. We analyse the effects of hand abstraction, then present and evaluate a hybrid solution to this problem. We present improvements in bluffing and straight and flush prediction. Finally, we extend BPP's opponent modeling.

1. INTRODUCTION

Poker is an ideal vehicle for testing automated reasoning under uncertainty. Poker is a game intermediate between chess and chance: there are important opportunities for the exercise of strategic planning, tactical skills and the astute observation and learning of the abilities and tendencies of opponents; and there are important elements of uncertainty, making all of learning, planning and execution difficult. Uncertainty is introduced through the physical randomness of shuffling, the incomplete information available about an opponent's cards and the limited information available to construct psychological models of opponents and their playing behaviours. For poker is famously a game of psychology — effective modeling of one's opponents and effective bluffing are critical for success. Poker is a game where competing agents must perform estimation, prediction, risk management, deception and opponent modelling. Finally, poker is a game all about betting, and betting behaviour is the foundation and origin of our most powerful formalization for coping with uncertainty, namely probability theory, as well as our most powerful philosophical theory of uncertainty, namely Bayesianism (Ramsey, 1931). All this being the case, it seems apt to apply Bayesian network technology to automated poker playing.

Bayesian networks (BNs) are now a well-established tool in the Artificial Intelligence community for reasoning under uncertainty. A BN is a graph with arrows between nodes and no loops (technically, a directed acyclic graph), whose nodes represent random variables and whose arcs represent direct dependencies. Each node has associated with it a conditional probability table (CPT), which, for each combination of the values of the parent nodes, gives a probability of each value of the child variable. Users can specify the states of any combination of nodes in the network. This evidence propagates through the network, producing a new probability distribution over all the variables in the network. There are a number of efficient exact and approximate inference algorithms for performing this probabilistic updating (Pearl, 1988), providing a powerful combination of predictive, diagnostic and explanatory reasoning. Decision networks, also known as Influence Diagrams (Howard and Matheson,

¹email:ann.nicholson@infotech.monash.edu.au

²email:kevin.korb@infotech.monash.edu.au

1981), are extensions to BNs which support decision making, by explicitly representing the decisions under consideration and the cost/benefits, or more generally utility of the resultant outcomes. By combining probabilistic reasoning with utilities, decision networks help us make decisions that maximize the expected utility.

There has been a range of other work on automated poker play. Findler (1977) was the first to work on automated poker play, using a combination of a probabilistic assessment of hand strength with the collection of frequency data for opponent behaviour to support the refinement of the models of opponent. Waterman (1970) and Smith (1983) used poker as a testbed for automatic learning methods, specifically the acquisition of problem-solving heuristics through experience. Koller and Pfeffer (1997) have developed *Gala*, a system for automating game-theoretic analysis for two-player competitive imperfect information games, including simplified poker. Although they use comparatively efficient algorithms, the size of the game tree is too large for this approach to be applied to full poker. Most recently, Billings et al. (in a series of work from 1995, summarised well in (Billings *et al.*, 2002)) have investigated the automation of Texas Hold'em poker with their program *Poki*.

Our previous work on Bayesian and decision networks for 5-card stud poker has appeared both in the refereed research literature (Korb, Nicholson, and Jitnah, 1999; Korb and Nicholson, 2004) and in unpublished honours theses (e.g. (Jitnah, 1993; Thomson, 1995; Carlton, 2000; Boulton, 2003)). Throughout this paper, we make comparisons where required to this previous Bayesian poker work. Our 5-card stud BPP was evaluated experimentally against two automated opponents: (1) a probabilistic player that estimates its winning probability for its current hand by taking a large sample of possible continuations of its own hand and its opponent's hand, then making its betting decision using the same method as BPP; (2) a simple rule-based opponent that incorporated plausible maxims for play (e.g., fold when your hand is already beaten by what's showing of your opponent's hand). 5-card stud BPP was also tested against earlier versions of itself to determine the effect of different modeling choices. Finally, 5-card stud BPP has been tested against human opponents with some experience of poker who were invited to play via telnet. In all cases, we used BPP's cumulative winnings as the evaluation criterion. The most recent 5-card stud version of BPP (as described in (Korb and Nicholson, 2004, Section 5.3)) performed significantly better than both the automated opponents, was on a par with average amateur humans, but lost fairly comprehensively to an expert human poker player.

In Section 2, we describe poker, and specifically the variant addressed here, Texas Hold'em. In Section 3 we present a Bayesian decision network which compactly represents two-person Texas Hold'em poker, provides an estimate of the probability of winning after each card is dealt and uses this to compute the betting action that will maximize the expected utility. In Section 4 we show how we use betting curves to randomise the betting actions and in Section 5 we describe a hybrid approach to overcome some of the inaccuracies introduced by hand abstraction. Then we describe the further extensions to BBP including an improved bluffing strategy (Section 6), flush and straight prediction from partial hands for Texas Hold'em (Section 7) and extended opponent modelling (Section 8). Finally, we discuss some likely ways of improving the program (Section 9).

2. POKER: THE GAME

Poker is an indeterministic imperfect information betting game that is played all over the world. There are many forms of poker, however, they all follow the same basic rules. In general, a game is contested between two and ten competitors, using a standard deck of fifty-two playing cards. At the beginning of each game, each player is required to contribute a small opening bet or *ante* to a communal collection called the *pot*. Each player is then dealt five or more cards, one at a time, and attempts to use these cards to create the strongest poker hand possible. In most formats of the game, some cards are known as *hole cards* and are seen only by the player, while other cards are dealt face up and visible to all players. Certain five card combinations are recognized in all forms of poker and the ranking of these combinations gives a poker hand its strength. These are described, from weakest to strongest, in Table 2.

Poker is a betting game, with players betting that they will end up with the strongest five-card hand, thus winning the pot containing all bets. The bets occur in well defined betting rounds. A betting round begins with the player to the left of the dealer and proceeds in a clockwise direction. At each turn, a player may take one of three courses of action: **Raise**, where a player may place a bet that exceeds the amount of the last bet; **Call**, where a player may match the last bet previously placed by another player; and **Fold**, where, in the event that another player has placed a bet exceeding the player's last bet, a player may discard ones cards and forfeit any previous contributions to the pot. In this case, the player takes no further part in the hand.

Hand Type	Example	Probability	
		5 card stud	Texas Hold'em
Busted	A♣ K♠ J♦ 10♦ 4♥	0.5015629	0.1728400
Pair	2♥ 2♦ J♠ 8♣ 4♥	0.4225703	0.4380000
Two Pair	5♥ 5♣ Q♠ Q♣ K♣	0.0475431	0.2351900
Three of a Kind	7♣ 7♥ 7♠ 3♥ 4♦	0.0211037	0.0483400
Straight (sequence)	3♠ 4♣ 5♥ 6♦ 7♠	0.0035492	0.0479900
Flush (same suit)	A♣ K♣ 7♣ 4♣ 2♣	0.0019693	0.0299000
Full House	7♠ 7♦ 7♣ 10♦ 10♣	0.0014405	0.0255000
Four of a Kind	3♥ 3♠ 3♦ 3♣ J♠	0.0002476	0.0018800
Straight Flush	3♠ 4♠ 5♠ 6♠ 7♠	0.0000134	0.0003600

Table 1: Poker hand types: weakest to strongest with prior probabilities in both Five Card Stud and Texas Hold'em

The fact that an opponent can fold, without seeing the contents of a player's hand, enables the possibility of *bluffing*, in which a player may bet aggressively on a weak hand hoping to scare away the (likely stronger) competition. A betting round concludes when either: (1) a player makes a bet that no other active player is willing to match (and is entitled to collect the entire pot with out revealing the contents of his or her hand); (2) all active players call the highest bet and the game then continues and more cards are dealt before another betting round.

After all cards have been dealt, a final betting round is conducted. If this process does not determine a winner, the game proceeds to a showdown and the pot goes to the active player with the strongest poker hand. Hands in the same category are distinguished by the rank of their most important cards. For instance, a pair of kings outranks a pair of fours. In the event of two players having a pair of kings, the winner is decided by the rank of their next highest card. If these are also equal, the comparison continues to the next highest ranking card until a winner is determined. If the two hands are in fact equal, the pot is shared evenly. Suits are not considered except when identifying a flush or straight flush. A session of poker typically consists of several games between the same group of players, allowing each player to learn the different strategies employed by their opponents and to adapt accordingly. In each new game, the player to the left of the previous dealer becomes the new dealer.

Five-card stud was the format used in our earlier Bayesian poker work. After an ante is paid, each player is dealt two cards, one down (hidden) and one up (open), and a betting round is conducted. Three further betting rounds follow, each after an additional up card is dealt. Finally, there is a showdown if more than one player remains.

Texas Hold'em has become the game of choice for recent research (as well as TV!), such as the work done by Billings et al. (e.g. (2002)). It is a variation of stud poker in which the open cards are shared by all players. These common cards are dealt out initially, but revealed only on the *flop*, when betting occurs. Limit Hold'em includes a constraint on bet sizes, namely that each bet must be of either a large or small denomination, the large typically being twice the value of the small. The game begins with the player to the left of the dealer paying an ante, called the small blind, which is equal to half the value of a small bet. The player to their left is required to pay the large blind, equivalent to a small bet. The game then proceeds to the *pre-flop* where each player is dealt two hole cards face down. A betting round then occurs. The next stage is called the *flop* during which three *table* cards are dealt face up in the center of the table, available to all players. A further betting round is then conducted. Two more rounds then occur, called the *turn* and *river* respectively, in which a single card is added to the table and a further betting round occurs. Bets in the first two rounds must be of the small value while bets in the final two rounds must be of the large value. A limit of three raises in any round must be adhered to. If more than one player remains active at the end of the river, the game proceeds to a showdown. Each active player is then required to create the strongest five-card poker hand from their two hole cards and the five table cards, and these hands decide the winner of the pot.

The basic decision facing any poker player is to estimate one's winning chances accurately, taking into account how much money will be in the pot if a showdown is reached and how much it will cost to reach the showdown. Assessing the chance of winning is not simply a matter of the probability that the hand you have now, will end up stronger than your opponent's hand, if the five table cards are dealt out. Such a pure combinatorial probability of winning is clearly of interest, but it ignores a great deal of information that good poker players rely upon. It

ignores the “tells” some poker players have (e.g., facial tics, fidgeting); it also ignores current opponent betting behaviour and the past association between betting behaviour and hand strength. Our Bayesian Poker Player (BPP) doesn’t have a robot’s sensory apparatus, so it can’t deal with tells, but it does account for current betting behaviour and learns from the past relationship between opponents’ behaviour throughout the game and their hand strength at showdowns.

3. A BAYESIAN DECISION NETWORK FOR TEXAS HOLD’EM POKER

The network shown in Figure 1 models the relationship between current hand type, final hand type, the behaviour of the opponent and the betting action. Note that all the standard BN variable nodes are represented with an oval shape, while the diamond *Winning* node is a utility node and the rectangular node *BPP_Next_Action* is a so-called “decision” node. Unlike earlier versions of BPP structure, where each round was modelled by a separate network, here the variations required for each of the rounds of play are modelled through the influence of the *Round* node, providing continuity between the various phases of a hand.

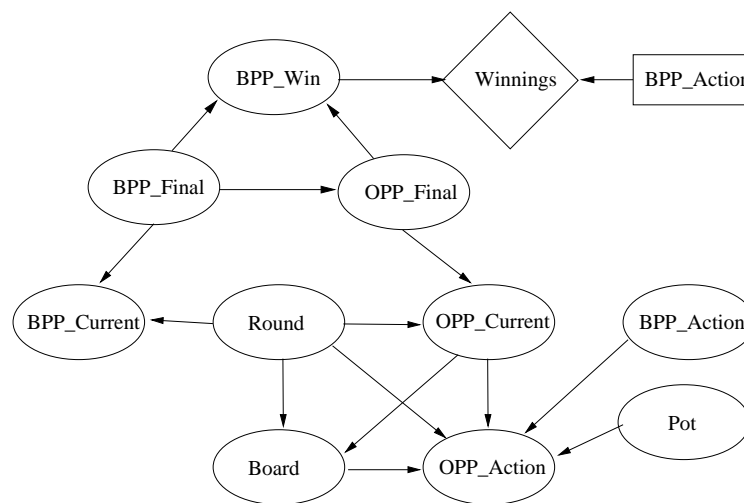


Figure 1: A decision network for poker.

The node *OPP_Final* represents the opponent’s final hand type, while *BPP_Final* represents BPP’s final hand type; that is, these represent the hand types they will have after all five cards are dealt. Whether or not BPP will win is the value of the Boolean variable *BPP_Win*; this will depend on the final hand types of both players. *BPP_Final* is an observed variable after the final card is dealt, whereas its opponent’s final hand type is observed only after play ends in a showdown. Note that the two final hand nodes are *not* independent, as one player holding certain cards precludes the other player holding the same cards; for example, if one player has four-of-a-kind aces, the other player cannot.

The nodes representing hand types are given values which sort hands into strength categories. In principle, we could provide a distinct hand type to each distinct poker hand by strength, since there are finitely many of them. That finite number, however, is fairly large from the point of view of Bayesian network propagation; for example, there are already 156 differently valued Full Houses. The current version of BPP recognizes 25 types of hand, subdividing busted hands into busted-low (8 high or lower), busted-medium (9, 10 or J high), busted-queen, busted-king and busted-ace, representing each paired hand separately, and with the 7 other hand types. The investigation of different refinements of hand types is described below Section 5.

At any given stage, BPP’s current hand type is represented by the node *BPP_Current* (an observed variable), while *OPP_Current* represents its opponent’s current hand type. Since BPP cannot observe its opponent’s current hand type, this must be inferred from the information available: the board at a particular stage of the game, represented by the *Board* node,³ and the opponent’s actions, represented by node *OPP_Action*.

The CPT $P(\text{Board} | \text{BPP_Current}, \text{OPP_Current})$ give the conditional probabilities of a given hand showing on

³Note that the *Board* node replaces the two nodes *OPP_Upcards* and *BPP_Upcards* found in the 5 card stud network structure.

the table when the current hands are of a certain type. The remaining CPTs are giving the conditional probability for each type of partial hand given that the final hand will be of a particular kind, used for both *OPP_Current* and *BPP_Current*. These CPTs were estimated by dealing out 10,000,000 hands of poker.

This network assumes an opponent’s action depends on its current hand, the value of the pot at the time when the opponent was considering their action (*Pot* node) and BPP’s most recent betting action (*BPP_Past_Action*). This extends the opponent model in our previous work, where the (obviously too simplistic) assumption was made that the opponent’s action depended only on its current hand. We note that the network still does not model such things as the opponent’s confidence or bluffing strategy. The nodes representing the both the opponent’s and BPP’s actions (past and next) have eight possible values, *fold*, *call*, *raise*, *check*, *bet*, *paysmallblind*, *paylargeblind*, *pass*. This is a considerable advancement on previous BPP’s limited action choices for 5-card stud poker, where betting actions were folded into 3 alternatives *bet/raise*, *pass/call*, *fold*. The *Pot* node values are discretised into 10 states that represent 20 dollar increments in value.

The conditional probability table for the *OPP_Action* node holds the model of the opponent by mapping a situation in the game to a probability distribution over these possible actions. The information that influences this node includes: the opponent’s current hand (25 states), the board (25 states), the round (4 states), the value of the pot (10 states) and the previous action of the player (8 states). This gives a CPT with 200,000 entries. Considering that a typical game of poker may only consist of 100 to 1000 hands of poker, it is unlikely that significant samples will be observed to meaningfully update this CPT during an encounter with a particular opponent. This aspect is considered further in Section 5 below. We also note that since the rules of poker do not allow the observation of hidden cards unless the hand is held to showdown, any adjusting of these entries are made only for such hands, undoubtedly introducing some bias.

Given evidence for *BPP_Current*, *Board*, and *OPP_Action*, belief updating produces belief vectors for both players’ final hand types and, most importantly, a posterior probability of BPP winning the game.

Given an estimate of the probability of winning, it remains to make betting decisions. Recall that decision networks can be used to find the optimal decisions which will maximize an expected utility. For BPP, the decision node *BPP_Next_Action* in Figure 1 represents the possible betting actions, which are the same eight alternatives as for the *OPP_Action* node, while the utility we wish to maximize is the amount of winnings BPP can accumulate.

The utility node, *Winnings*, measures the dollar value BPP expects to make based on the possible combinations of the states of the parent nodes (*BPP_Win* and *BPP_Next_Action*). For example, if BPP decided to fold with its next action, irrespective of whether or not it would have won at a showdown, the expected future winnings will be zero as there is no possibility of future loss or gain in the current game. On the other hand, if BPP had decided to bet and it were to win at a showdown, it would make a profit equal to the size of the final pot F_{bet} , minus any future contribution made on its behalf B_{bet} . If BPP bet and lost, it would make a loss equal to any future contribution it made towards the final pot, $-B_{bet}$. A similar situation occurs when BPP decides to pass, but with a differing expected total contribution B_{pass} and final pot F_{pass} . This information is represented in a utility table within the *Winnings* node, shown in Table 2.

<i>BPP_Next_Action</i>	<i>BPP_Win</i>	Utility
Bet	Win	$F_{bet} - B_{bet}$
Bet	Lose	$-B_{bet}$
Pass	Win	$F_{pass} - B_{pass}$
Pass	Lose	$-B_{pass}$
Fold	Win	0
Fold	Lose	0

Table 2: Poker action/outcome utilities

The amount of winnings that can be made by BPP is dependent upon a number of factors, including the number of betting rounds remaining R , the size of the betting unit B and the current size of the pot C . The expected future contributions to the pot by both BPP and OPP must also be estimated (see (Carlton, 2000) for details).

The decision network then uses the belief in winning at a showdown and the utilities for each (*BPP_Win*, *BPP_Next_Action*) pair to calculate the expected winnings (EW) for each possible betting action. Folding is always considered to have zero EW, since regardless of the probability of winning, BPP cannot make any future loss or profit.

4. BETTING WITH RANDOMIZATION

The decision network presented in the previous section provides a “rational” betting decision, in that it determines the action that will maximize the expected utility if the showdown is reached. However, if a player invariably bets strongly given a strong hand and weakly given a weak hand, other players will quickly learn of this association; this will allow them to better assess their chances of winning and so to maximize their profits at the expense of the more predictable player. So BPP employs a mixed strategy that selects an action with some probability based on the EW of the action. This ensures that while most of the time BPP will bet strongly when holding a strong hand and fold on weak hands, it occasionally chooses a locally sub-optimal action, making it more difficult for an opponent to construct an accurate model of BPP’s play.

Betting curves, such as that in Figure 2, are used to randomize betting actions. The horizontal axis shows the difference between the EW of folding and calling⁴ (scaled by the bet size); the vertical axis is the probability with which one should fold. Note that when the difference is zero ($EW(call) - EW(fold) = 0$), BPP will fold randomly half of the time.

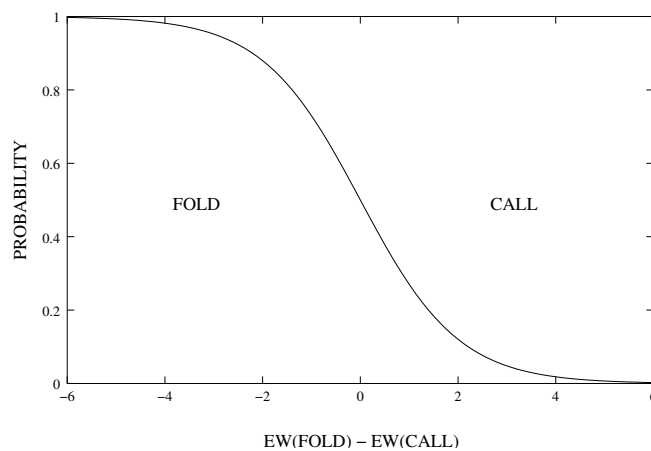


Figure 2: Betting curve for folding.

Once the action of folding has been rejected, a decision needs to be made between calling and raising. This is done analogously to deciding whether to fold, and is calculated using the difference between the EW of betting and calling.

The betting curves were generated with exponential functions, with different parameters for each round of play. Ideal parameters will select the optimal balance between deterministic and randomized play by stretching or squeezing the curves along the horizontal axis. If the curves were stretched horizontally, totally random action selection could result, with the curves selecting either alternative with probability 0.5. On the other hand, if the curves were squeezed towards the center, a deterministic strategy would ensue, with the action with the greatest EW always being selected. The current parameters in use by BPP were obtained for 5-card stud using a stochastic search of the parameter space when running against an earlier version of BPP. We expect that a further search to optimise these values for the current BPP playing Texas Hold’em will provide further small improvement.

5. A HYBRID APPROACH

5.1 The limitations of hand abstraction

One of the obvious disadvantages to BPP’s play is the so-called “hand abstraction” used to contain the complexity of the decision network. In our various versions of 5-card stud BPP we tinkered with different abstractions,

⁴More exact would be to compute the differential EW between folding and not folding, the latter requiring a weighted average EW for *pass/call* and for *bet/raise*. We use the EW of calling as an approximation for the latter. Note also that we refer here to calling rather than passing or calling, since folding is not a serious option when there is no bet on the table, implying that if folding is an option, passing is not (one can only pass when there is no bet).

from 9 to the current 25 hand types (which still represented over 133 million card combinations). The hand abstraction causes some information to be lost and therefore introduces inaccuracy into the model; in particular, subtle differences between similar hands, that are crucial when determining which hand will win, may be lost. A balance between model performance and computational requirements is required. Here we analyse the magnitude of the inaccuracy in this trade-off.

Two measures of the hand abstraction inaccuracy are considered. Firstly, a theoretical measure, the Kullback-Leibler divergence (Cover and Thomas, 1991), is considered as a measurement of inaccuracy between the *BPP_Win* node probability distributions output by two different models. Secondly, a more practical measure is observed by playing different models against each other, and also against the automated Texas Hold'em player *poki* developed by Billings et al.⁵

In addition, a so-called *Combinatorial* model was designed as a base case against which other models could be tested. Each position in a game is evaluated by considering 100,000 random decks of cards. This number of trials was chosen as the largest number that would allow acceptable playing speed using the resources available. The hand is dealt out to completion using these decks and frequency counts are compiled for winning, tied, and losing outcomes. This information is used to calculate an accurate probability that the agent will hold the strongest hand if the game should proceed to a showdown. No information is lost in the hand representation for this model, meaning that it does not suffer from the hand abstraction problems effecting the BPP network. Also, this method does not consider information concerning the opponent's actions. When comparing BPP to the Combinatorial model, no evidence is added about the opponent's actions, which allows a direct comparison of the effects of hand abstraction alone.

Five models with differing granularities of hand abstraction (33, 25, 18, 15, 13) were compared to determine the effect that the number of states has on the accuracy of the network (see (Boulton, 2003) for details of this experiment). The busted and pair categories were focused on due to the high frequency with which they occur, thus making this experiment more sensitive. Each model was compared with the combinatorial model in each of the four betting rounds to gauge how they performed through out the entire course of a hand. In each category, the KL divergence was calculated by averaging the results from 100,000 random hands. These results are shown graphically in Figure 3.

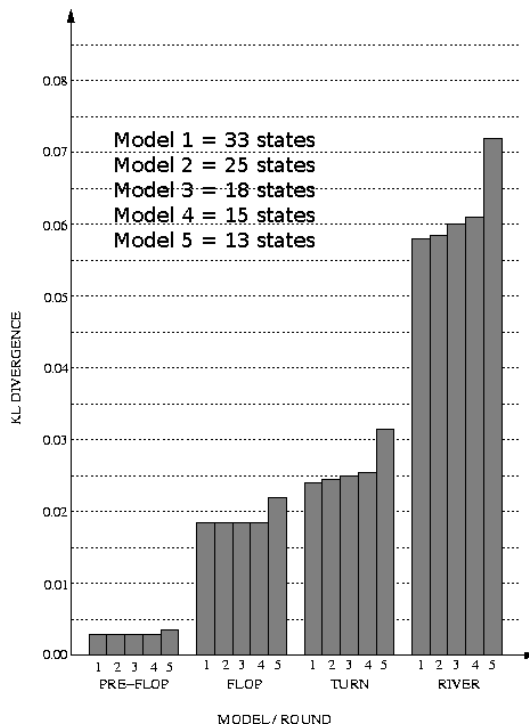


Figure 3: The KL divergence between BN model and the Combinatorial model for 5 different abstractions.

⁵Note that all results described here were obtained against the 2003 version of *Poki*.

From Figure 3, it is clear that hand abstraction has a much larger influence in the later betting rounds, while being almost irrelevant at the beginning of a hand. Secondly, it appears that initial decreases in granularity cause a significant increase in accuracy, while further decreases have less effect. This is especially true when it is considered that model 1 (33 states) requires a relatively greater increase in state numbers for a much smaller gain in accuracy. Our current 25 hand types seems a reasonable compromise. Table 2 indicates that hands with two pairs becomes much more common when playing the game of Texas Hold'em. This suggests that this hand type might make a good candidate for further segmentation when future increases in state numbers are considered.

While these theoretical results confirm that hand abstraction does indeed play an important role, they do not quantify the actual disadvantage that it will cause under practical playing conditions. The performance of an automated agent in actual game play is clearly the gold standard for testing successful strategies. To this end, the primitive Bayesian Model with no opponent modeling was played against the Combinatorial Model over 20,000 random hands to quantify the practical financial loss that was occurring. As these two models differ only in the presence of hand abstraction, the losses observed by the Bayesian Model can be directly credited to this factor. Model 2 (25 states) was used for this experiment. This model was observed to lose 0.49 ± 0.12 small betting units per hand against the Combinatorial model.

To further evaluate this effect, both models were evaluated against the Billing's agent over the online poker server. Due to the speed of play over the Internet, smaller numbers of hands could be played. Over 6000 hands, the Bayesian Model lost 0.43 ± 0.17 small betting units per hand. Over the same number of hands, the Combinatorial model lost 0.33 ± 0.15 small betting units per hand. It is interesting to note that this difference (0.1 small betting units per hand) is smaller than the losses observed when the two models are played against each other. One possible cause for this discrepancy might be the learning techniques incorporated into the Billing's agent. These techniques may be more efficient against the Combinatorial model, thus reducing the gap between them. These results demonstrate that hand abstraction provides a significant financial disadvantage in situations of actual game play and is a worthwhile area in which to invest further experimental resources. We now present one possible solution to this problem.

5.2 Incorporating the combinatorial models probability estimates

Both the Combinatorial model and the decision network model have advantages and disadvantages when implemented as a poker playing agent. The Combinatorial model, while being able to provide accurate predictions of final hand probabilities, is poorly equipped to implement an effective opponent modeling strategy. The decision network model, while providing the potential for intuitive and transparent opponent modeling, suffers from the effects of hand abstraction, meaning that it is unable to predict the final hand distributions accurately. The Hybrid model aims to meld these two techniques in an attempt to provide accurate mathematical predictions of final hand types while still allowing effective opponent modeling.

First, in the hybrid model, the priors for each final hand are adjusted to the result from the Combinatorial model. This required the separation of the link between the two final hand nodes, but it was felt that this introduced only a small inaccuracy. In a similar fashion, this technique was intended to bias the results of the BN towards that of the Combinatorial model. By itself, however, this doesn't work, as, intuitively, information from the various sources of available evidence are effectively introduced twice, causing an overly extreme belief to be computed. In self-play trials, this model fared significantly worse than the unaltered BN.

In order to rectify this problem, additional uncertain, or so-called "virtual" evidence was entered into the two final hand nodes. This uncertain evidence represented the difference between the original BN probabilities and those produced by the Combinatorial roll outs. This was done by using the ratio by which the distribution estimated by the combinatorial method differed from the original BN estimate. These values were then normalised and re-introduced to the network⁶. The details of this method rely on an understanding of the message passing algorithm used to perform BN inference, and is beyond the scope of this paper. The interested reader is referred to (Korb and Nicholson, 2004)[Section 3.4] for a description of the use of virtual evidence, and (Boulton, 2003) for a detailed description of the hybrid method implementation.

The Hybrid model was evaluated against both the Combinatorial model and the original decision network model (as per Section 3). The Hybrid model was able to win 0.50 ± 0.12 small betting units per hand from the BN model over 20,000 hands. This was a similar result to that achieved by the Combinatorial model. As expected,

⁶Using the Netica(2000) BN software's ``EnterNodeLikelihood`` function

the Hybrid model lost a small amount against the Combinatorial model (0.02 small betting units per hand), although this was not statistically significant. The Hybrid model shows that BNs are capable of being combined with more accurate techniques to almost eliminate the effects of hand abstraction. We use the Hybrid model as the base model for the experiments with other extensions to BPP in the following section.

6. BLUFFING

Bluffing is the intentional misrepresentation of the strength of one’s hand. You may over-represent that strength (what is commonly thought of as bluffing), in order to chase opponents with stronger hands out of the round. You may equally well under-represent the strength of your hand (“sandbagging”) in order to retain players with weaker hands and relieve them of spare cash. These are tactical purposes behind almost all (human) instances of bluffing. On the other hand, there is an important strategic purpose to bluffing, as von Neumann and Morgenstern pointed out, namely “to create uncertainty in [the] opponent’s mind” (von Neumann and Morgenstern, 1947, pp. 188-189). In BPP this purpose is already partially fulfilled by the randomization introduced with the betting curves. However, that randomization occurs primarily at the margins of decision making, when one is maximally uncertain whether, say, calling or raising is optimal over the long run of similar situations. Bluffing is not restricted to such cases; the need is to disguise from the opponent what the situation is, whether or not the optimal response is known. Hence, bluffing is desirable for BPP as an action in addition to the use of randomizing betting curves.

In our earlier work (Korb *et al.*, 1999), BPP bluffed (by over-representation) in the last round of betting with a low probability (5%). In a subsequent version of BPP (described in (Korb and Nicholson, 2004)), BPP used the notion of a “bluffing state.” First, BPP worked out what its opponent will believe is BPP’s chance of winning, by performing belief updating given evidence for *BPP_Upcards*, *OPP_Upcards* and *OPP_Action*. Given this belief is non-zero, it is worth considering bluffing. In which case BPP had a low probability of entering the bluffing state in the last round of betting, whereupon it continued to bluff (by over-representation) until the end of the round. In addition, several rules were introduced to prevent obviously foolish behaviour. For example, the agent would not bluff if the opponent had enough information to know that the agent held a losing hand. Once in bluffing mode, the agent would continue to raise for the remainder of the hand, until the opponent folded or a showdown was reached. It was felt that this method of bluffing did not mimic the behaviour of the agent in cases where a strong hand was actually held. The reason for this stems from the random betting strategy employed by the agent. In cases where a strong hand is held, the agent will generally raise, but the random betting strategy will cause the agent to occasionally call, and rarely to fold. This causes the continual raising behaviour of the bluffing agent to be easily distinguished from the cases when it actually holds a strong hand. In order to prevent the opponent learning this strategy, a more natural bluffing behaviour was required.

Here we describe a simple modification was made to this bluffing algorithm. Instead of automatically raising once in the bluffing mode, BPP instead now over-represents the strength of its hand. The decision network was used to produce a predicted probability of winning by the usual method. This value was then modified to halve the chance of losing. For instance, an predicted 60 percent chance of winning would be modified to 80 percent. This value was then used to calculate the expected utilities in the same way as before, and the remaining calculations proceeded as normal. In this way, the agent would behave in a way consistent with holding a stronger hand than in reality.

A limited trial (5000 hands of Hybrid model with static opponent modelling (see Section 8 below) with each of the old and new bluffing strategies) was undertaken against the online Billings agent, which incorporates an effective opponent modeling strategy and would be expected to take advantage of the predictable behaviour of the original bluffing algorithm. The results given in Table 3. A small but not significant improvement (in betting units won/lost per hand) was observed; further experiments are required to confirm the improvement obtained by the modified bluffing strategy.

Trial	Outcome
Hybrid + Old Bluffing vs Billings	Billings won 0.243 sbu/hand
Hybrid + New Bluffing vs Billings	Billings won 0.238 sbu/hand

Table 3: Summary of the results for modified bluffing. sbu = small betting unit

7. IMPROVED PREDICTION FROM PARTIAL HANDS

The poker decision network already includes the ability to recognise when a straight or flush is being currently held, as these hand types are represented by a state in the current and final hand nodes. However, during the course of a game, it is also important to recognise partially formed straights and flushes in order to predict the probability that these hand types will eventuate in future rounds. The original model simplified by represented these partial hand types directly in the final and current hand nodes. However, in Texas Hold'em the overlap between such partial hands and legitimate final hands (e.g., pairs) will become far more significant due to the increased frequency of straights and flushes in Texas Hold'em compared to Five Card Stud; for example, straights occur 13.5 times more frequently while flushes occur 15.1 times more often in this form of the game. Therefore, a more sophisticated method for representing the potential for future straights and flushes is required. Many possible solutions to this problem exist but in this instance, three additional nodes were added to the network representing the partial flushes present in the board, the opponent's current hand and the player's current hand. Partial flushes were represented as the highest number of cards of the same suit present in a particular hand. The relevant fragment of this extended decision network structure is shown in Figure 4.

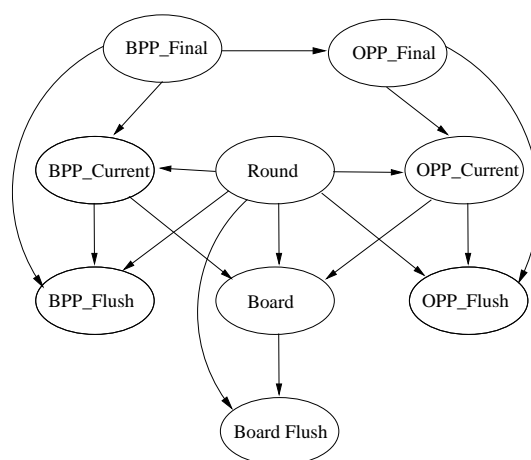


Figure 4: Modified network fragment showing the ability to represent partial flushes in the board, opponent's current hand and the player's current hand.

This new network structure was evaluated against the unmodified version which was identical except for the changes described above. Over 20,000 hands of poker, the new model was able to outperform the previous version at the rate of 0.128 ± 0.123 small betting units per hand. This demonstrates that the ability to recognise partial flushes is important to successful poker play, especially in forms of the game that deliver these hands to players at significant frequencies. Representing straights is more complex because these types of hands can occur in many combinations with missing cards occurring at the beginning, middle or end of an already held sequence. Further work to determine an optimal selection of states, however a similar modification to the flush case could represent partial straight combinations and would be expected to provide a similar improvement.

8. OPPONENT MODELLING

8.1 Static Opponent Modelling

Static opponent modeling refers to a technique in which an agent's model of an opponent does not change during a game. Instead, the agent uses a generic opponent model that is thought to represent the strategy of reasonable opponent and employs this model against all opponents. There are a vast number of differing strategies used in poker. This means that a static opponent model will never be able to effectively deal with each different style that it encounters. However, this form of opponent modeling still has a valuable role, both in the initial stages of a game when little information on the current opponent is known and also in situations where useful techniques to model an opponent are not available.

Here we describe a new technique for initialising the *Opp_Action* CPT with useful probability distributions that reasonably model a generic opponent.

The technique begins with the CPT filled with uniform distributions. From this point, an updating process is repeatedly employed to adjust this table until it represents a generic opponent. Each round of updating builds upon the results of the last in a process resembling the bootstrapping of a computer. Each round of updating involves considering every possible combination of parent states. For each combination, the BN itself is used to compute the probability distribution over each possible action. These probabilities are then inserted back into the CPT that models the opponent's actions. Before the first round of updating, the CPT contains only even distributions meaning that the BN is in effect ignoring the opponent's actions. By inserting these results back into the CPT, the BN now assumes that it is playing an opponent that itself ignores their opponent's previous actions. After the second round, a situation arises in which the BN is now assuming that their opponent considers that its opponent is not considering the previous actions of their opponent. In a two handed game, the opponent of a player's opponent is obviously the player themselves but there is no theoretical reason why this technique could not be adapted to multiplayer games. Since there are only finite actions in each betting round, the actions further back in the game will cause less and less change to a player's beliefs. This means that as each round of updating is performed, the rate of change in the CPT will decrease. To measure this change, the mean squared difference (MSD) between the CPT for successive rounds was measured (see (Boulton, 2003) for detailed results). We found that after 6 rounds of updating, further updating makes very little difference. For this reason, it was decided that 6 rounds would be used for bootstrap initialisation of future versions of the network.

During two-handed games of poker over 20,000 hands, the model using this static opponent modeling technique has been shown to outperform an identical model without this feature. The magnitude of this advantage was shown to be 2.14 ± 1.09 small betting units per hand. It was also shown to outperform the Combinatorial model by 1.12 ± 1.07 small betting units per hand over 20,000 hands.

8.2 Dynamic Opponent Modelling

Dynamic opponent modeling refers to techniques in which the opponent model is continually updated during a game to more accurately reflect the strategy of a specific opponent. This provides the flexibility to adapt effectively to the vast variety of playing styles against which an agent is likely to compete. Unfortunately, the main obstacle to effectively implementing dynamic opponent modeling stems from the enormous number of situations in which an agent must accurately be able to predict its opponent. As indicated earlier, in BPP's decision network, the CPT for the *OPP_Action* contains 200,000 entries, a number far too great to allow for effective learning within the short duration of typical poker game. Thus the challenge lies in discovering methods by which to accelerate the learning process to a point where it becomes beneficial within the space of 100 poker hands (a reasonable duration for a professional poker game between humans).

There are several methods by which the CPT might be reduced in size. Most obviously, a smaller number of parents will decrease the number of entries significantly. An increase in the granularity (i.e. reduce the number of states) will achieve a similar result. Unfortunately, this will also decrease the accuracy of the opponent model. Instead, it was decided to reduce the number of values to be learned by segmenting the CPT into groups of situations where a reasonable player might make similar actions. For example, in certain situations, a player might fold if the pot is less than 80 dollars, combining the cases where the pot is less than this. Evidence observed in one situation can be also used to learn other situations. The interest lies in finding a method to effectively segment the CPT. As the CPT becomes segmented into fewer groups, the learning rate will increase at the expense of accuracy. It is important to find an appropriate compromise between these two important considerations, and this is likely to change depending on the length of game in which the agent is expected to compete. The two source of information use are: (1) hand information when a showdown occurs; and (2) BPP's estimate of the opponents hidden cards during the game.

We chose to use a decision tree to represent the CPT (a method used elsewhere, e.g. (Friedman and Goldszmidt, 1996)). In particular, we applied a decision tree learner D-Graf,⁷ that uses Minimum Message Length (MML) (Wallace, 2005), trading off the learning rate with the accuracy. MML has the advantage that it provides theoretically optimal results for many learning tasks.

The first implementation of this method learned the structure of the tree offline from generic data that was ob-

⁷This is a MML decision tree program developed internally at Monash University.

served from actions made by the Hybrid model during self-play trials. This aimed to provide a tree structure that was as generic as possible and best equipped to model a large variety of different playing styles. The data was actually simulated by using the Hybrid model to calculate a probability distribution over each possible action, for each situation represented by the 200,000 parent combinations for the opponent action node. This distribution was used to create 100 observations in ratios that reflected the probability of each action. The 20,000,000 observations were then processed by D-Graf to produce a decision tree. During a game, this tree was used to update the CPT from its original values that were inherited from the static opponent model (see above). As each observation becomes available during a game, the decision tree is used to allocate that observation to a leaf. The program then passes through each entry in the CPT, updating all those that also lie in that leaf.

To date we have undertaken only a limited experimentation with this dynamic opponent modelling method, investigating the effect of varying (1) the learning rate and (2) the size of the decision tree used to segment the CPT. Preliminary results from a small number of trials between the hybrid model with and with dynamic opponent modelling (reported in (Boulton, 2003)) indicate that the method has some potential. However much more research and experimentation is required to establish that the method improves BPP's play against automated players using extensive opponent modelling and, of course, human players.

9. CONCLUSION

Poker is the quintessential game combining physical probabilities (the randomness introduced by shuffling) with epistemic probabilities (the unknown values of hidden cards) with the uncertainties of assessing the opponent's psychology (propensity to bluff, strategic intentions). In previous work, we showed a Bayesian approach can play 5-card stud poker reasonably well. In this paper we have successfully adapted our approach to the variant, Texas Hold'em poker, including special handling of flush and straight prediction required by the shared board cards.

We have also presented modifications applicable to any poker version, including: (1) modifying BPP's bluffing to mimic betting for a stronger hand; (2) using a hybrid model that incorporates estimates from a combinatorial hand roll out to overcome some of the limitations of using hand abstractions; and (3) developing new static and dynamic opponent modelling methods. We have presented results comparing versions of BPP with each other, with the combinatorial model, and with the Billings online Texas hold'em poker player. In all cases, Billings beat BPP, however there is much room for improving BPP's Texas Hold'em play, for example, by searching for new betting curve parameters and improved dynamic opponent modelling, especially by modelling an opponent's bluffing. Further improvement can be achieved by modelling the interrelation between rounds of play using a so-called dynamic decision network.

In the future we would like to make the poker-playing environment more challenging, in particular introducing multi-opponent games and allowing table stakes games. In multiple opponent games it will be more important to incorporate the interrelations between what is known of different player's hands and the node representing their final hands. In table stakes any player may bet or raise as much money as is jointly available at the time to that player and any remaining opponent. This makes the precise computation of winning probabilities rather more critical. Thus, table stakes poker provides a much more severe test environment than fixed-size betting.

Acknowledgements

We acknowledge the initial work by Nathalie Jitnah and the assistance at various stages of the project of Jon Oliver, Tam Lien, Aidan Doyle, Jason Carlton, Jamie Scuglia and Scott Thompson.

10. REFERENCES

- Billings, D., Davidson, A., Schaeffer, J., and Szafron, D. (2002). The challenge of poker. *Artificial Intelligence*, Vol. 134, pp. 201–40.
- Boulton, D. (2003). Bayesian Poker. Honours thesis, School of Computer Science and Software Engineering, Monash Univ. <http://www.csse.monash.edu.au/hons/projects/2003/Darren.Boulton/>.

- Carlton, J. (2000). Bayesian Poker. Honours thesis, School of Computer Science and Software Engineering, Monash University. <http://www.csse.monash.edu.au/hons/projects/2000/Jason.Carlton/>.
- Cover, T. M. and Thomas, J. A. (1991). *Elements of Information Theory*. Wiley-Interscience.
- Findler, N. (1977). Studies in machine cognition using the game of poker. *Communications of the ACM*, Vol. 20, pp. 230–245.
- Friedman, N. and Goldszmidt, M. (1996). Learning Bayesian networks with local structure. *UAI96 – Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence* (eds. Horvitz and Jensen), pp. 252–262.
- Howard, R. and Matheson, J. (1981). Influence Diagrams. *Readings in Decision Analysis* (eds. R. Howard and J. Matheson), pp. 763–771. Strategic Decisions Group, Menlo Park, CA.
- Jitnah, N. (1993). Bayesian Poker. Honours thesis, Dept. of Computer Science, Monash University.
- Koller, D. and Pfeffer, A. (1997). Representations and solutions for game-theoretic problems. *Artificial Intelligence*, Vol. 94, pp. 167–215.
- Korb, K. B. and Nicholson, A. E. (2004). *Bayesian Artificial Intelligence*. Computer Science and Data Analysis. Chapman & Hall / CRC, Boca Raton.
- Korb, K. B., Nicholson, A. E., and Jitnah, N. (1999). Bayesian poker. *UAI99 – Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (eds. Laskey and Prade), pp. 343–350, Sweden.
- Neumann, J. von and Morgenstern, O. (1947). *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ, second edition.
- Norsys (2000). Netica. <http://www.norsys.com>.
- Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, CA.
- Ramsey, F. P. (1931). Truth and Probability. *The Foundations of Mathematics and Other Essays* (ed. R. Braithwaite). Humanities Press, New York.
- Smith, S. F. (1983). Flexible learning of problem solving heuristics through adaptive search. *IJCAI-83*, pp. 422–425.
- Thomson, S. (1995). Bayesian Poker. Honours thesis, Dept of Computer Science, Monash Univ.
- Wallace, C. (2005). *Statistical and Inductive Inference by Minimum Message Length*. Information Science and Statistics. Springer, New York.
- Waterman, D. A. (1970). A generalization learning technique for automating the learning of heuristics. *Artificial Intelligence*, Vol. 1, pp. 121–170.