Successful Performance via Decision Generalisation in No Limit Texas Hold'em

Jonathan Rubin and **Ian Watson**
Department of Computer Science
University of Auckland, New Zealand
jrubin@gmail.com, ian@cs.auckland.ac.nz

Abstract

Given a set of data, recorded by observing the decisions of an *expert* player, we present a case-based framework that allows the successful generalisation of those decisions in the game of no limit Texas Hold'em. We address the problems of determining a suitable **action abstraction** and the resulting **state translation** that is required to map real-value bet amounts into a discrete set of abstract actions. We also detail the similarity metrics used in order to identify similar scenarios, without which no generalisation of playing decisions would be possible. We show that we were able to successfully generalise no limit betting decisions from recorded data via our agent, SartreNL, which achieved a 5th place finish out of 11 opponents at the 2012 Annual Computer Poker Competition.

1 Introduction

In 2008 the Second Man-Machine Poker Competition was won by a computer poker robot named Polaris [Bowling *et al.*, 2009]. Polaris challenged a group of professional human players to the game of limit Texas Hold'em, beating its competitors by a statistically significant margin. The success of Polaris can (at least in part) be attributed to the increasing popularity of Texas Hold'em poker as a research domain and advances in game theoretic equilibrium finding algorithms [Sandholm, Winter 2010; Rubin and Watson, April 2011]. Polaris's victory occurred in the game of **limit** Texas Hold'em, where the amount able to be wagered is restricted by pre-determined bet sizes. Today, the most popular variation of the game is **no limit** Texas Hold'em, where players' bet sizes are no longer restricted. This allows a player to wager any amount they wish (up to the total amount of chips they possess). This simple rule change has a profound effect on the nature of the game, as well as on the development of computerised agents that wish to handle the no limit betting structure. While the Polaris system was able to beat world-class human opposition in the game of limit Hold'em, nothing like Polaris's victory has been achieved in the more complicated domain of no limit Hold'em. In fact, it is only relatively recently that many researchers have shifted their attention to the more complicated domain of no limit Texas Hold'em [Schnizlein *et al.*, 2009; Van den Broeck *et al.*, 2009; Gilpin *et al.*, 2008].

Our previous research has focused on the techniques of expert imitation to achieve strong performance in the domain of limit Texas Hold'em [Rubin and Watson, 2010]. In this current work, we apply the same principles of expert imitation and decision generalisation to the more complicated domain of no limit Texas Hold'em. For automated no limit poker agents, a non-trivial translation phase is required to map quantitative bet amounts into discrete betting categories. Furthermore, our approach requires the construction of metrics to determine similarity between complicated no limit betting sequences (without which no generalisation would be able to take place). Our approach allows the successful imitation of any designated *expert player* (artificial or human), given a large enough set of training data. We present results from the 2012 Annual Computer Poker Competition (ACPC), where our entry to the competition, SartreNL, achieved a 5th place finish in the no limit equilibrium run-off event [1].

2 Background

In [Rubin and Watson, 2010] we focused on expert imitation via a case-based approach within the domain of limit Texas Hold'em. Expert decisions recorded from a set of training data were encoded into cases and playing decisions were made at runtime by searching a case-base. While we employ a similar framework for our current work, the transition to a no limit domain results in unique challenges that are not encountered in a limit poker environment. First, there is the issue of establishing a set of abstract betting actions that all real actions will be mapped into during game play. This is referred to as **action abstraction** and it allows the vast, continuous domain of no limit Hold'em to be approximated by a much smaller *abstract* state space. Second, given an established set of abstract actions, a **translation** process is required that determines how *best* to map real actions into their appropriate abstract counterparts, as well as a **reverse translation** that maps abstract actions back into appropriate real-world betting decisions.

Both **action abstraction** and **state translation** are issues that are also required to be addressed in the construction of no

[1] http://www.computerpokercompetition.org

Table 1: The case representation used by SartreNL. The four features capture important game state information. A solution is made up of an action and outcome tuple.

Feature	Type	Example
1. Hand Strength Bucket	Integer	1 – 50
2. Betting Sequence	String	*pdc-cqc-c*, *cc-*, *dc-qc-ci*, ...
3. Stack Commitment	Integer	1,2,3,4
4. Board Texture	Class	*No-Salient*, *Flush-Possible*, *Straight-Possible*, *Flush-Highly-Possible*, ...
Action	n-tuple	(0.0, 1.0, 0.0, 0.0, ...), ...
Outcome	n-tuple	(-∞, 36.0, -∞, -∞, ...), ...

limit ϵ-Nash equilibrium strategies via algorithmic game theory. A pair of strategies are said to be an ϵ-Nash equilibrium if either player cannot gain more than ϵ by deviating their strategy. An early attempt to construct a no limit Hold'em agent via game theoretic methods is described by Andersson [Andersson, 2006]. Andersson extended the procedures used to construct ϵ-Nash equilibrium-based limit poker agents [Billings *et al.*, 2003] to the no limit domain.

Another no limit poker agent produced via game theoretic algorithms is Tartanian [Gilpin *et al.*, 2008]. Tartanian was able to build models for much larger starting chip stacks than the work presented by [Andersson, 2006].

While our current work is required to deal with many of the same issues and challenges faced by ϵ-Nash Equilibrium strategies, the focus of our approach is more to do with expert imitation and investigating the generalisation of playing decisions from game traces.

3 Overview

Given a set of (artificial or real-world) training data, our agent SartreNL, is able to generalise the decisions recorded within the data by constructing and storing a collection of cases. Each case attempts to capture important game state information that is likely to have an impact on the final betting decision. Table 1 depicts a collection of attribute-value pairs that, when taken together, captures a particular game scenario. All attributes were selected by the authors, given their importance in determining a final betting decision.

Each case also records a solution. A solution is made up of two n-tuples, one which specifies action probabilities and another which specifies the average outcome of taking the observed action in the past. The entries within each tuple correspond to a particular betting decision. The entries within the action tuple must sum to one.

During game play values are assigned to each attribute and the previously stored collection of cases are searched for attributes with similar values. The case with the highest global similarity is assumed to be most similar to the current situation. Once a similar case has been retrieved a betting decision is made by re-using the tuples within that case's solution.

Each attribute-value pair is described in more detail below:

Hand Strength Bucket This is given by *rolling out* all possible combinations of community cards and determining the proportion of the time the player's hand wins against the set of all possible opponent holdings. Values are mapped into a discrete set of buckets that contain hands of similar strength.

Betting Sequence The betting sequence attribute is given by a string of characters where each character is an observed game action. As any betting value can be observed in the real game, a discrete set of abstract actions are chosen to represent real betting actions.

Stack Commitment The proportion of chips committed by a player, compared to the player's stack size. The stack commitment feature maps this value into one of N categories, where N is a specified granularity.

Board Texture The board texture attribute maps a collection of community cards to one of nine categories that highlight important information such as whether a straight or flush is possible.

4 Action Abstraction

Recall that *abstraction* is a concept used by game theoretic poker agents that derive ϵ-Nash equilibrium strategies for the game of Texas Hold'em. As the actual Hold'em game tree is much too large to represent and solve explicitly, it becomes necessary to impose certain abstractions that help restrict the size of the original game. For Texas Hold'em, there are two main types of abstraction:

1. **Chance abstraction** – which reduces the number of chance events that are required to be dealt with. This is typically achieved by grouping strategically similar hands into a restricted set of buckets.
2. **Action abstraction** – which restricts the number of actions a player is allowed to perform.

A typical abstraction such as: *fcpa*, restricts the allowed actions to: *f* – fold, *c* – call, *p* – bet the size of the pot *a* – all-in (i.e. the player bets all their remaining chips). Given this abstraction, all actions are interpreted by assigning the actual actions into one of their abstract counterparts.

While SartreNL does not attempt to derive an ϵ-Nash equilibrium solution for no limit Hold'em, it is still required to define an action abstraction in order to restrict the number of actions allowed in the game and hence reduce the state space. SartreNL uses the following action abstraction: *fcqhipdvta*, where each entry refers to different discrete betting categories as follows: fold (*f*), call (*c*), raise quarter pot (*q*), raise half pot (*h*), raise three quarter pot (*i*), raise pot (*p*), raise double pot (*d*), raise five times pot (*v*), raise ten times pot (*t*), all-in (*a*).

5 Translation

Given that all bets need to be mapped into abstract actions, a translation process is required to define the appropriate mapping. [Schnizlein *et al.*, 2009] formalise two types of translation: hard translation and soft translation.

- **Hard Translation:** is a many to one mapping that maps an unabstracted betting value into an abstract action based on a chosen distance metric. Given a unique unabstracted betting value, hard translation will always map this value into the same abstract action. A disadvantage of hard translation is that an opponent can exploit this mapping simply by selecting particular betting values.
- **Soft Translation:** is a probabilistic state translation that uses normalised weights as similarity measures to map an unabstracted betting value into an abstract action. The use of a probabilistic mapping ensures that soft translation cannot be exploited like hard translation can.

SartreNL uses both hard and soft translation. The type of translation that takes place differs depending on where translation occurs within the system. The exact details of the translation used within the different areas of the system are now presented.

Define $A = \{q, h, i, p, d, v, t, a\}$ to be the set of abstract betting actions. Note that the actions f and c are omitted as these require no mapping.

1. During case-base construction, where hand history logs from previously played hands are encoded into cases, SartreNL uses the hard translation function $T_h : \Re \to A$, as follows:

$$T_h(b) = \begin{cases} a & \text{if } \frac{a}{b} > \frac{b}{c} \\ c & \text{otherwise} \end{cases} \quad (1)$$

   where $b \in \Re$ is the proportion of the total pot that has been bet in the actual game and $a, c \in A$ are abstract actions that map to actual pot proportions in the real game and $a <= b < c$. The fact that hard translation has the capability to be exploited is not a concern during case-base construction. Hard translation is used during this stage to ensure that re-training the system with the same hand history data will result in the same case-base.

2. During actual game play SartreNL is required to map opponent betting actions (as well as its own actions) to abstract categories. Observant opponents have the capability to exploit deterministic mappings during game play, hence SartreNL uses a soft translation function for this stage, $T_s : \Re \to A$, given by the following probabilistic equations:

$$P(a) = \frac{\frac{a}{b} - \frac{a}{c}}{1 - \frac{a}{c}} \quad (2)$$

$$P(c) = \frac{\frac{b}{c} - \frac{a}{c}}{1 - \frac{a}{c}} \quad (3)$$

   where once again, $b \in \Re$ is the proportion of the total pot that has been bet in the actual game and $a, c \in A$ are abstract actions that map to actual pot proportions in the real game and $a <= b < c$. Note that when $b = a, P(a) = 1$ and $P(c) = 0$ and when $b = c, P(a) = 0$ and $P(c) = 1$. Hence, a betting action that maps directly to an abstract action in A does not need to be probabilistically selected. On the other hand, when $b \neq a$ and $b \neq c$, abstract actions are chosen probabilistically. Also, notice that in Equations (2) and (3), $P(x) + P(y) \neq 1$ and hence a final abstract action is probabilistically chosen by first normalising these values.

3. The final place that translation is required is when SartreNL has determined an appropriate abstract action to play. A reverse mapping is then required to map the abstract action into an appropriate real betting value, given the current game conditions. SartreNL uses the following function to perform reverse translation, $T_r : A \to \Re$:

$$T_r(a) = a' \pm \Delta a' \quad (4)$$

   where $a \in A$ and $a' \in \Re$ is the real value corresponding to abstract action a and $\Delta a'$ is some random proportion of the bet amount that is used to ensure SartreNL does not always map abstract actions to their exact real world counterparts. For example, when $a' = 100$ and $\Delta = 0.3$, SartreNL could bet any amount between 70 - 130 chips.

6 Similarity

In order to generalise no limit betting decisions, it is first required to locate similar scenarios for which solutions have been recorded in the past. Given a target case, t, that describes the immediate game environment, a source case, $s \in S$, where S is the entire collection of previously recorded cases and a set of features, F, global similarity is computed by summing each feature's local similarity contribution, sim_f, and dividing by the total number of features:

$$G(t, s) = \sum_{f \in F} \frac{sim_f(t_f, s_f)}{|F|} \quad (5)$$

We now present the local similarity metrics (sim_f) required in order to generalise betting decisions from a collection of data.

Table 2: Bankroll instant run-off results. This table is replicated from the 2012 ACPC

	Round 0	Round 1	Round 2	Round 3	Round 4	Round 5	Round 6	Round 7	Round 8	Round 9
1. Hyperborean	586 ± 30	527 ± 28	530 ± 31	273 ± 23	223 ± 25	167 ± 27	167 ± 31	181 ± 37	174 ± 24	161 ± 36
2. Tartanian5	597 ± 27	536 ± 27	509 ± 30	159 ± 30	97 ± 27	-7 ± 26	17 ± 29	4 ± 34	-76 ± 27	-161 ± 36
3. NeoPokerLab	534 ± 23	456 ± 23	424 ± 25	198 ± 23	86 ± 26	16 ± 28	-58 ± 31	-78 ± 32	-97 ± 23	–
4. Little Rock	1116 ± 27	929 ± 28	907 ± 30	298 ± 26	128 ± 25	37 ± 27	-24 ± 35	-107 ± 35	–	–
5. SartreNL	112 ± 25	19 ± 25	-21 ± 29	56 ± 22	-57 ± 22	-97 ± 23	-101 ± 27	–	–	–
6. Hugh	483 ± 18	459 ± 19	422 ± 22	71 ± 18	-49 ± 17	-117 ± 18	–	–	–	–
7. Spewy Louie	355 ± 28	257 ± 29	204 ± 33	-230 ± 31	-427 ± 31	–	–	–	–	–
8. Lucky7_12	-809 ± 57	-719 ± 47	-1009 ± 52	-826 ± 45	–	–	–	–	–	–
9. Azure Sky	-196 ± 65	-1209 ± 60	-1966 ± 68	–	–	–	–	–	–	–
10. dcubot	-1097 ± 15	-1254 ± 16	–	–	–	–	–	–	–	–
11. UNI-mb	-1681 ± 47	–	–	–	–	–	–	–	–	–

6.1 Hand Strength Bucket

The following metric is used to determine similarity between two hand strength buckets (f_1, f_2).

$$sim(f_1, f_2) = max\{1 - k \cdot \frac{|f_1 - f_2|}{T}, 0\} \quad (6)$$

Here, T refers to the total number of buckets that have been defined, where $f_1, f_2 \in [1, T]$ and k is a scalar parameter used to adjust the rate at which similarity should decrease. SartreNL uses values of $T = 50$ and $k = 2$.

6.2 Stack Commitment

The stack commitment metric uses an exponentially decreasing function.

$$sim(f_1, f_2) = e^{(-|f_1 - f_2|)} \quad (7)$$

where, $f_1, f_2 \in [1, N]$ and N refers to the granularity used for the stack commitment attribute. This function was chosen as small differences between two stack commitment attributes (f_1, f_2) should result in large drops in similarity. SartreNL uses a granularity of $N = 4$.

6.3 Betting Sequence

For two betting sequences to be considered similar each sequence must contain the same number of elements and any calls (c) or all-in bets (a) that occur within sequence S_1 must also occur at the same location in sequence S_2. Exactly how dissimilar two individual bets are to each other can be quantified by how far away from each other they occur within the bet discretisation string: *qhipdvt*, e.g. $\delta(h, i) = 1$ and $\delta(q, t) = 6$.

For two betting sequences S_1, S_2 overall similarity is determined by (8):

$$sim(S_1, S_2) = \begin{cases} 1 - \sum_{i=0}^{|S_1|} \delta(S_{1,i}, S_{2,i})\alpha & \text{if } |S_1| = |S_2|, \\ & S_{1,i} = c \Rightarrow S_{2,i} = c, \\ & S_{1,j} = a \Rightarrow S_{2,j} = a \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

where the notation $S_{1,i}, S_{2,i}$ refers to the i^{th} character in the betting sequences S_1 and S_2, respectively and α is some constant rate of decay.

6.4 Board Texture

To determine similarity between board texture categories a similarity matrix of hand picked values was derived by the authors.

7 Results

A version of the system described above was submitted to the 2012 Annual Computer Poker Competition. Our entry to the competition was trained on data from the best no limit agent of the 2011 competition. The ACPC is the premier computer poker event and has been held each year at either AAAI or IJCAI conferences since 2006. Entrance into the competition is open to anyone and the agents submitted typically represent the current state of the art in computer poker.

Table 2 presents a cross-table of results between competitors at the 2012 heads-up no limit competition. A green cell indicates a win for the row player and a red cell indicates a loss for that player. Cells with a lighter background represent matches that were not statistically significant. The figures presented in Table 2 are in milli big blinds per hand (mb/h), where the total number of big blinds won are divided by the number of hands played, and the result is multiplied by 1000.

Table 2 presents the final results of the instant run-off competition. The instant run-off competition uses a recursive winner determination algorithm that repeatedly removes the agents that performed the worst against a current pool of players.

8 Conclusion

Our results support the idea that generalising decisions via expert imitation has the ability to produce strong, sophisticated strategies in complex, imperfect information environments. Table 2 shows that these strategies can lead to successful performance compared with alternative approaches. In particular, the SartreNL system produced by the framework achieved a 5th place finish out of 11 competitors at the 2012 ACPC no limit, instant run-off competition.

References

[Andersson, 2006] Rickard Andersson. Pseudo-optimal strategies in no-limit poker. Master's thesis, Umea University, 2006.

[Billings *et al.*, 2003] Darse Billings, Neil Burch, Aaron Davidson, Robert C. Holte, Jonathan Schaeffer, Terence Schauenberg, and Duane Szafron. Approximating game-theoretic optimal strategies for full-scale poker. In *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, pages 661–668, 2003.

[Bowling *et al.*, 2009] Michael Bowling, Nicholas Abou Risk, Nolan Bard, Darse Billings, Neil Burch, Joshua Davidson, John Hawkin, Robert Holte, Michael Johanson, Morgan Kan, Bryce Paradis, Jonathan Schaeffer, David Schnizlein, Duane Szafron, Kevin Waugh, and Martin Zinkevich. A demonstration of the polaris poker system. In *AAMAS '09: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems*, pages 1391–1392, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems.

[Gilpin *et al.*, 2008] Andrew Gilpin, Tuomas Sandholm, and Troels Bjerre Sørensen. A heads-up no-limit texas hold'em poker player: discretized betting models and automatically generated equilibrium-finding programs. In *7th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, pages 911–918, 2008.

[Rubin and Watson, 2010] Jonathan Rubin and Ian Watson. Similarity-based retrieval and solution re-use policies in the game of texas hold'em. In *Lecture Notes in Computer Science, 2010, Volume 6176. Case-Based Reasoning. Research and Development, 18th International Conference on Case-Based Reasoning.*, pages 465–479. Springer-Verlag, 2010.

[Rubin and Watson, April 2011] Jonathan Rubin and Ian Watson. Computer poker: A review. *Artificial Intelligence*, 145(5-6):958–987, April 2011.

[Sandholm, Winter 2010] Tuomas Sandholm. The state of solving large incomplete-information games, and application to poker. *AI Magazine*, Winter, 2010.

[Schnizlein *et al.*, 2009] David Schnizlein, Michael H. Bowling, and Duane Szafron. Probabilistic state translation in extensive games with large action sets. In *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 278–284, 2009.

[Van den Broeck *et al.*, 2009] Guy Van den Broeck, Kurt Driessens, and Jan Ramon. Monte-carlo tree search in poker using expected reward distributions. In *Advances in Machine Learning, First Asian Conference on Machine Learning, ACML 2009*, pages 367–381, 2009.