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
jrub001@aucklanduni.ac.nz, ian@cs.auckland.ac.nz
http://www.cs.auckland.ac.nz/research/gameai

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. The transition from a **limit** betting structure to a **no limit** betting structure offers challenging problems that are not faced in the limit domain. In particular, 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 2nd place finish at the 2010 Annual Computer Poker Competition.

1 Introduction

In 2008 the Second Man-Machine Poker Competition was won by a computer poker robot named Polaris [1]. 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 [2, 3]. 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 [4–6].

Our previous research has focused on the techniques of expert imitation to achieve strong performance in the domain of limit Texas Hold'em [7]. 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. As mentioned above, while the transition from restricted betting to a no limit betting structure ostensibly is a simple rule change, the result is a profound impact on the nature of the game, as well as on the construction of computerised agents. For automated no limit poker agents, a non-trivial translation phase is now 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 describe the discrete betting categories used by our system, the translation process that maps real values into appropriate categories and the metrics required that allow successful generalisation of an expert player's decisions based on a set of hand histories.

Despite the extra complications introduced by the no limit Hold'em domain, we show that our approach is still able to achieve successful performance. In particular, we present results from the 2010 Annual Computer Poker Competition (ACPC), where our entry to the competition, SartreNL, achieved a 2nd place finish in the no limit equilibrium run-off event [8].

In Section 2 we describe the game of no limit Texas Hold'em. Section 3 provides the necessary background for the current work. Section 4 provides an overview of our approach. Sections 5 and 6 focus on how no limit betting is discretised and on the non-trivial translation phase that maps quantitative betting values into their discretised categories. Section 7 introduces the metrics required in order to generalise an expert player's observed decisions and Section 8 lists results obtained from the 2010 ACPC followed by a discussion of these results. Finally, a conclusion is provided in Section 9.

2 No Limit Texas Hold'em

Here we briefly describe the game of Texas Hold'em, highlighting some of the common terms which are used throughout this work. We focus on 2-player no limit Hold'em, as our system has been specialised for this domain. When a game consists only of two players, it is described as a **heads-up** match.

The game of heads-up, no limit Texas Hold'em is played in 4 stages – **preflop**, **flop**, **turn** and **river**. During the preflop both players are dealt two **hole cards**, which only they can see. Before any betting takes place, two forced bets are contributed to the pot, i.e. the **small blind** and the **big blind**. The big blind is typically double that of the small blind. The possible betting actions common to all variations of poker are described as follows:

- **Fold:** When a player contributes no further chips to the pot and abandons their hand and any right to contest the chips that have been added to the pot.
- **Check/Call:** When a player commits the minimum amount of chips possible in order to stay in the hand and continue to contest the pot. A check requires a commitment of zero further chips, whereas a call requires an amount greater than zero.
- **Bet/Raise:** When a player commits greater than the minimum amount of chips necessary to stay in the hand. When the player could have checked, but decides to invest further chips in the pot, this is known as a bet. When the player could have called a bet, but decides to invest further chips in the pot, this is known as a raise.

In a **limit** game all bets are in increments of a certain amount. However, in a **no limit** game a player may bet any amount up to the total value of chips that they possess. In a standard game of heads-up, no-limit poker both players' chip stacks would fluctuate between hands, e.g. a win from a previous hand would ensure that one player had a larger chip stack to play with on the next hand. In order to reduce the variance that this structure imposes, a variation known as **Doyle's Game** is played where the starting stacks of both players are reset to a specified amount at the beginning of every hand.

Once the betting is complete, as long as no player has folded, play continues on to the next stage. Each further stage involves the drawing of **community cards** from the shuffled deck of cards as follows: **flop** – 3 community cards, **turn** – 1 community card, **river** – 1 community card.

During each stage, players combine their hole cards with the public community cards to form their best 5 card poker hand. Each stage involves a further round of betting. A **showdown** occurs after the river where the remaining players reveal their hole cards and the player with the best hand wins all the chips in the pot. If both players' hands are of equal value, the pot is split between them.

3 Background

Our previous work has focused on expert imitation via a case-based approach within the domain of limit Texas Hold'em [7]. Expert decisions recorded from a set of training data are encoded into cases. Playing decisions are then made at runtime by searching the 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, quantitative 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 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 [9]. Andersson extended the procedures used to construct ϵ-Nash equilibrium-based limit poker agents [10] to the no limit domain. Betting amounts were discretised based on the amount of chips currently in the pot. Four abstract actions were created: half the pot, the full amount in the pot, 2 $\times$ the pot and all-in (all the player's remaining chips). All bet amounts were required to be mapped into one of the above four abstract actions. Andersson notes that simply assigning a bet amount into the abstract action with the closest absolute distance can result in strategies that are able to be exploited. Instead Andersson advocates a probabilistic mapping based on the inverse absolute distance between abstract actions. Due to processor and memory limitations, [9] was only able to produce a game theoretic strategy for very small starting chip stacks.

Another no limit poker agent produced via game theoretic algorithms is Tartanian [6]. Tartanian was able to build models for much larger starting chip stacks than the work presented by [9]. Gilpin et. al. [6] advocate a translation process that uses a relative distance to perform the mapping, rather than an absolute distance. Using relative distance, a bet amount that falls between two abstract actions is mapped into the appropriate action by considering their corresponding ratios.

State translation in extensive form games has been formalised by Schnizlein et. al. [4]. Schnizlein et. al. define the concepts of **hard translation** and **soft translation**. Hard translation refers to the deterministic mapping of bet amounts into their appropriate categories. Soft translation refers to the probabilistic mapping of these values. Schnizlein et. al. investigate both hard and soft translation in the domains of Texas Hold'em poker and Leduc Hold'em — a much smaller, specialised poker domain useful for experimental analysis. In both domains [4] show that the use of hard translation produces strategies that are more easily exploitable than soft translation.

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.

We are now in a position to present the main components that make up our system. The resulting no limit poker agent is referred to as SartreNL. We begin with an overview of the representation used to record game scenarios by processing a collection of training data. The exact *action abstraction* used by SartreNL and the details of how and where *state translation* occurs are described, along with the metrics that allow decision generalisation to take place.

4 Overview

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, -∞, -∞, ...), ...

Given a set of (artificial or real-world) training data, 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. SartreNL uses four attribute-value pairs to describe the current state of a match. Three of the four attributes (*hand strength*, *betting sequence*, *board texture*) are the same as those used by the limit variation of Sartre [7]. The *stack commitment* attribute was introduced especially for the no limit variation of the game. 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:

4.1 Hand Strength Bucket

To evaluate the strength of a player's hand the *expected hand strength squared* metric is used $E[HS^2]$. The $E[HS^2]$ metric computes the probability of winning at showdown against a random hand. 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. The hand strength value for each community card *roll-out* is then squared and the final $E[HS^2]$ is given by averaging these values.

Given the large variety of values that can be produced by the $E[HS^2]$ metric, bucketing takes place where similar values are mapped into a discrete set of buckets that contain hands of similar strength. SartreNL uses a total of 50 buckets for each post-flop betting round.

4.2 Betting Sequence

The betting sequence attribute is given by a string of characters where each character is either an observed game action or round delimiter. Round delimiters are represented by hyphens and indicate the transition to the next round of betting. As any quantitative betting value can be observed in the real game, a discrete set of abstract actions are chosen to represent real betting actions. This is known as *action abstraction* and is described in more detail in Section 5. The action abstraction used by SartreNL is given in Table 2.

Betting sequences consist of every abstract action that was observed up until the current decision point in the game. This includes a player's own previous actions and actions for all previous rounds of play. By including actions that belong to previous rounds, similarity assessment is made more difficult, but this allows a more informed context about the game environment at the time a playing decision was made.

4.3 Stack Commitment

In the no limit variation of Texas Hold'em players can wager any amount up to their total stack size. The proportion of chips committed by a player, compared to the player's stack size, is therefore of much greater importance, compared to limit Hold'em. The betting sequence maps bet amounts into discrete categories based on their proportion of the pot size. This results in information that is lost about the total amount of chips a player has contributed to the pot, relative to the size of their starting stack. Once a player has contributed a large proportion of their stack to a pot, it becomes more important for that player to remain in the hand, rather than fold, i.e. they have become **pot committed**.

The **stack commitment** feature maps this value into one of N categories, where N is a specified granularity:

$$[0 - \frac{1}{N}], [\frac{1}{N} - \frac{2}{N}], \ldots, [\frac{N-2}{N} - \frac{N-1}{N}][\frac{N-1}{N} - 1]$$

Hence, for a granularity of $N = 4$, a stack commitment of 1 means the player has committed less than 25% of their initial stack, a stack commitment of 2 means that player has contributed somewhere between 25% up to 50% of their total stack, and so forth.

4.4 Board Texture

The **board texture** attribute highlights important information about the public community cards that all players share to make their best five card hand. For instance, on the last round of betting if four of the five community cards were all the same suit, the chances that a player's opponent has a flush is high as they only require one card of that suit within their personal hole cards. The board texture attribute maps a collection of community cards to one of nine categories. The categories were selected by the authors and are believed to distinguish between the salient aspects of the public community cards. The board texture categories are listed in Table 4.

5 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.

Action abstractions can typically be avoided by poker agents that specialise in limit poker, where there are only 3 actions to choose from: fold (f), check/call (c) or bet/raise (r). However in no limit, where a raise can take on any value, some sort of action abstraction is required. This is achieved by restricting the available bet/raise options to a discrete set of categories based on fractions of the current pot size. For example, 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*. Table 2 provides an explanation of the symbols used.

Table 2. The action abstraction used by SartreNL

f	fold
c	call
q	quarter pot
h	half pot
i	three quarter pot
p	pot
d	double pot
v	five times pot
t	ten times pot
a	all in

6 Translation

Given that all bets need to be mapped into one of the actions listed in Table 2, a translation process is required to define the appropriate mapping. For SartreNL, this translation phase needs to occur in 3 places:

1. During case base construction – where hand history logs from previously played hands are encoded into cases.
2. During actual game play - where betting actions observed during a hand are required to be mapped into appropriate abstract actions. Equivalent to the translation process required of ϵ-Nash equilibrium agents that solve an abstract extensive form game [9, 6, 4]
3. A final *reverse translation* phase is required to map a chosen abstract action into a real value to be used during game play.

Recall, Schnizlein et. al. [4] 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 SartreNL uses the hard translation function $T_h : \Re \rightarrow A$, 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, $T_s : \Re \rightarrow A$, 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.

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, $T_r : \ A \rightarrow \Re$:

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

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.

7 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:

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

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

7.1 Hand Strength Bucket

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

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

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 $k = 2$.

7.2 Stack Commitment

The stack commitment metric uses an exponentially decreasing function.

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

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 $N = 4$.

7.3 Betting Sequence

SartreNL uses the following bet discretisation: *fcqhipdvta*. Within this representation there are some non-identical bet sizes that are reasonably similar to each other. For example, a bet of half the pot (h) is quite close to a bet of three quarters of the pot (i). The betting sequence similarity metric we derived compares bet sizes against each other that occur at the same location within two betting sequences.

Let S_1 and S_2 be two betting sequences made up of actions $a \in A \cup \{f, c\}$, 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.

For two betting sequences to be considered similar they first need to satisfy the following conditions:

1. $|S_1| = |S_2|$
2. Both $S_{1,i} = c$ and $S_{1,j} = a$ whenever $S_{2,i} = c$ and $S_{2,j} = a$

i.e. each sequence contains 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[1].

Any two betting sequences that do not satisfy the initial two conditions above are assigned a similarity value of 0. On the other hand, if the two betting sequences do satisfy the above conditions their bet sizes can then be compared against each other and a similarity value assigned.

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, displayed in Table 3.

Table 3. Bet Discretisation String

q	h	i	p	d	v	t

As h and i are neighbours in the discretisation string they can be considered to occur at a distance of 1 away from each other, $\delta(h, i) = 1$, as opposed to say $\delta(q, t) = 6$, which are at opposite ends on the discretisation string.

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

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

where α is some constant rate of decay. SartreNL uses a rate of decay where $\alpha = 0.05$.

Betting Sequence Similarity Example Here we offer a concrete example of how similarity is computed for two non-identical betting sequences.

Consider two betting sequences, $S_1 = ihpc$ and $S_2 = dqpc$.

Here, $|S_1| = 4$ and $|S_2| = 4$ and wherever there exists a check/call (c) in S_1, there exists a corresponding c in S_2.

[1] A betting sequence consists of one or more betting rounds, the above conditions must be satisfied for all betting rounds within the betting sequence.

As both conditions are satisfied we can evaluate the top half of Equation (8):

$$\begin{aligned} sim(S_1, S_2) &= 1 - [\delta(i,d)\alpha + \delta(h,q)\alpha + \delta(p,p)\alpha + \delta(c,c)\alpha] \\ &= 1 - [2 \cdot \alpha + 1 \cdot \alpha + 0 \cdot \alpha + 0 \cdot \alpha] \\ &= 1 - 3\alpha \end{aligned}$$

Using a rate of decay of $\alpha = 0.05$, gives a final similarity of: $1 - 0.15 = 0.85$.

7.4 Board Texture

To determine similarity between board texture categories a similarity matrix was derived. Rows and columns in Figure 1 represent the different categories defined in Table 4. Diagonal entries refer to two sets of community cards that map to the same category, in which case similarity is always 1. Non-diagonal entries refer to similarity values between two dissimilar categories. These values were hand picked by the authors. The matrix given in Figure 1 is symmetric.

	A	B	C	D	E	F	G	H	I
A	1	0	0	0	0	0	0	0	0
B	0	1	0.8	0.7	0	0	0	0	0
C	0	0.8	1	0.7	0	0	0	0	0
D	0	0.7	0.7	1	0	0	0	0	0
E	0	0	0	0	1	0.8	0.7	0	0.6
F	0	0	0	0	0.8	1	0.7	0	0.5
G	0	0	0	0	0.7	0.7	1	0.8	0.8
H	0	0	0	0	0	0	0.8	1	0.8
I	0	0	0	0	0.6	0.5	0.8	0.8	1

Fig. 1. Board texture similarity matrix.

Table 4. Board Texture Key

A	No salient
B	Flush possible
C	Straight possible
D	Flush possible, straight possible
E	Straight highly possible
F	Flush possible, straight highly possible
G	Flush highly possible
H	Flush highly possible, straight possible
I	Flush highly possible, straight highly possible

8 Results

A version of the system described above was submitted to the 2010 Annual Computer Poker Competition [8]. Our entry to the competition was trained on data from the best no limit agent of the 2009 competition. The ACPC is the premier computer poker event and has been held each year at either AAAI or IJCAI conferences since 2006. The ACPC involves separate competitions for different varieties of Texas Holdem, such as limit and no-limit competitions, as well as heads-up and multiple-opponent competitions. Entrance into the competition is open to anyone and the agents submitted typically represent the current state of the art in computer poker.

Table 5. Crosstable of all matches. Results are from the perspective of the row player. Values are in milli big blinds per hand. Confidence intervals are omitted due to space considerations. This table is replicated from the 2010 ACPC [8]

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
(1) c4tw.iro	–	–	−4181	–	−7557	−2289	−5534	–
(2) c4tw.tbr	–	–	–	−3562	−6977	−2241	–	−8669
(3) Hyperborean.iro	4181	–	–	−83	775	200	248	122
(4) Hyperborean.tbr	–	3562	83	–	795	272	364	220
(5) PokerBotSLO	7557	6977	−775	−795	–	−193	−108	−159
(6) SartreNL	2289	2241	−200	−272	193	–	42	−13
(7) Tartanian4.iro	5534	–	−248	−364	108	−42	–	−80
(8) Tartanian4.tbr	–	8669	−122	−220	159	13	80	–

Table 6. Bankroll instant run-off results. This table is replicated from the 2010 ACPC [8]

	Round 0	Round 1	Round 2	Round 3
(1st) Hyperborean.iro	1351 ± 44	408 ± 27	224 ± 31	200 ± 39
(2nd) SartreNL	581 ± 34	12 ± 23	−79 ± 27	−200 ± 39
(3rd) Tartanian4.iro	1338 ± 33	−60 ± 27	−145 ± 34	–
(4th) PokerBotSLO	1620 ± 187	−359 ± 32	–	–
(5th) c4tw.iro	−4891 ± 213	–	–	–

Table 5 presents a cross-table of results between competitors at the 2010 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 5 are in milli big blinds per hand (mb/h), a milli big blind is 0.001 times

the big blind value. Therefore, mb/h are calculated by dividing the total number of big blinds won by the number of hands played, followed by multiplying the result by 1000. A result of +1000 mb/h means that a player won, on average, one big blind per hand.

In the no limit competition, the **Doyle's Game** rule variation was used where both player's begin each hand with 200 big blinds. All matches played were duplicate matches. In a heads-up duplicate match N hands are played between two agents after which the agents memories are wiped and the N hands played again, but in the reverse direction, i.e. the cards that were initially given to player A are instead given to player B and vice-versa. This way both players get to play both sets of N cards and this reduces the variance that is involved in simply playing a set of N hands in one direction only. Many duplicate matches are played in order to achieve a significant result. In the 2010 heads-up, no limit competition each competitor played 200 duplicate matches (each consisting of $N = 3000$ hands) against every other competitor.

Table 6 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. In the 2010 competition SartreNL was ranked in second place.

Table 5 indicates that SartreNL suffers a statistically significant loss against only one competitor i.e. Hyperborean. Where a competitor's name ends with .iro or .tbr this indicates the competitor was specifically submitted to a particular division (i.e. instant run-off or total bankroll). In general, SartreNL does not achieve as large a bankroll as some of the other competitors and this is mostly due to its performance against the competitor c4tw. It is clear that c4tw is the weakest competitor, having lost all of its matches, however SartreNL does not achieve as great a profit as some of the other competitors do against this opponent. Moreover, the amounts won against this opponent (the first two columns in Table 5) are by far larger than any other values in the table. This means that matches involving c4tw have a much greater impact on the final total bankrolls. This results in SartreNL achieving only a slight overall profit i.e. 581 ± 34. Table 6 shows that when c4tw is removed from the pool of players via the instant run-off winner determination procedure, SartreNL actually performs a lot better overall.

9 Conclusion

Given a set of data, recorded by observing an *expert* player, the framework presented in this paper allows the successful generalisation of those decisions. Our results support the idea that generalising decisions via expert imitation has the ability to produce strong, sophisticated strategies in complex, imperfect information environments. Moreover, our results show that these strategies can lead to successful performance compared with alternative approaches. In previous research we have shown this to be the case in the domain of limit Hold'em [7]. This work extrapolates our approach to the domain of no limit Hold'em. The

transition to a more complicated no limit betting structure required issues to be addressed that were not a concern within the limit domain. In particular, a suitable action abstraction was required in order to reduce the large no limit state space. Given a chosen abstraction, state translation needs to take place at various locations within the system. The formulas required for both hard and soft translation were explained, as were the local similarity metrics that allow the identification of similar scenarios so that decision generalisation can take place. The SartreNL system produced by the framework achieved a second place finish at the 2010 ACPC no limit, instant run-off competition.

References

1. 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.
2. Tuomas Sandholm. The State of Solving Large Incomplete-Information Games, and Application to Poker. *AI Magazine*, Winter, 2010.
3. Jonathan Rubin and Ian Watson. Computer poker: A review. *Artificial Intelligence*, 145(5-6):958–987, April 2011.
4. 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.
5. 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.
6. 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.
7. 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.
8. ACPC. The Annual Computer Poker Competition, 2011. `http://www.computerpokercompetition.org/`.
9. Rickard Andersson. Pseudo-Optimal Strategies in No-Limit Poker. Master's thesis, Umea University, 2006.
10. 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.