SPREE : The Strathclyde Poker Research Environment

Luke Dicken, Nicky Johnstone, John Levine and Phil Rodgers[1]

Abstract. This paper presents the Strathclyde Poker Research Environment (SPREE) which has been developed as a tool to aid research into the game of Poker, both by enhancing our understanding of the way human players play the game and providing a standard environment for autonomous agents to play. We will justify the need for this tool, outline the principal components and demonstrate potential uses for the system.

1 INTRODUCTION

1.1 Motivation

When designing and developing stronger Artificial Intelligence (AI) algorithms, researchers often look to classical multiplayer games as being microcosmic examples of the decision making process. Although games such as Go, Chess or Poker have simple rules, meaning that they can be described to an AI system relatively easily, they still remain complex challenges[7][4]. Of these, Poker is arguably one of the more interesting to AI as it a game with incomplete information; whereas with games such as Go and Chess all of the pieces are visible, and you can therefore make informed assumptions about your opponents' future moves, in Poker you have no knowledge of the cards your opponent is holding and must try to make inferences based on previous observations of that opponent which influence your decisions.

Poker research typically focuses on the "Texas Hold 'Em" variant, in which each player is dealt two cards which are hidden to other players, and subsequently a round of betting takes place based on the perceived strength of these "hole" cards. Three cards are then dealt face up and those players who still remain in the pot must now bet based on the strength of their two cards combined with the three on the table to make a five card Poker hand. When this is complete, a further card is dealt face up, followed by a round of betting, now based on the best five card hand each player can make from the six cards available to them. A final card is then dealt and a round of betting concludes the game, with those players still in the game revealing their hands and the player with the highest ranked hand taking the entire amount wagered by all players. This variant of Poker itself comes in one of two sub-variants, either "Limit" or "No-Limit". In both cases, players always have the option to "fold", forfeiting any wager made so far, and players may "call", or meet the amount currently being wagered by other players. They may also "raise", increasing the amount being bet. In Limit this raise is by a fixed amount, in No-Limit it is of a variable amount. Both are valuable mechanisms as Limit allows the researcher to deal with a more tractable problem, as only three actions are available at each decision point, whilst No-Limit allows a player to telegraph a lot more information about the perceived value of their hand which an AI system could use to influence its own decisions.

Many approaches to creating an automatic player (or "agent") for Poker have been based around the field of machine learning [3][6], using data about previous games to infer the likely strength of opponents, as well as decision theoretic techniques based on the observable components of the game. However, there is a common weakness to these systems in that both the quantity and quality of training data available from which to learn is not sufficient. Typically this data is obtained from players at online casinos, who observe and record the actions of their opponents using specialist software such as Poker Tracker[2], and subsequently trading or selling this data within the community. A sample training set, such as that used in a recent piece of work exploring the application of Monte Carlo tree search techniques to Poker play [5], consists of around a million games of Poker that have been observed. However, because this data has been collected by players at an online casino, only the information available to that player is recorded and able to be used by an AI system based on the data. As a result of this, the amount of information available - particularly about certain hand configurations - is quite limited. For instance, consider a player who is dealt a very bad hand initially. That player will probably fold immediately, and his hand will never be shown to the other players, so there is never an explicit connection (except in those cases where the player making the decision is the one recording the data) to link a poor initial hand to the fold action, it must instead be inferred from the frequency with which poor initial hands are seen at a showdown. Trying to rectify this large a priori bias in the dataset is the motivation for requiring such a large number of example games, but it does not address the fundamental issue that the partially observable nature of the game introduces.

Additionally, when such Poker agents are created by researchers they typically play Poker internal to themselves against either other agents created by the researchers (perhaps based on the work of others) or against a hypothetical model of a "human" player based on the data, rather than an actual human. The results of this play are then compared against other agents to look for improvements in performance according to some metric. The assumption underlying this kind of analysis is that for a large enough sample of games, two agents will experience an equivalent set of circumstances and thus be comparable. However, if you assume a full ten player game of Poker, there are 20 hole cards to be dealt and 5 community cards, and the number of possible permutations for dealing 25 cards from a deck is $7.41\text{x}10^{39}$, and this is before you consider the differences introduced by the behaviour of each agent's opponents when faced with the same situation. As such, it is effectively impossible to offer a meaningful comparison between two agents on the basis of independent experimentation with contrasted results, as the agents will not have experienced identical circumstances.

The Strathclyde Poker Research Environment (SPREE) aims to

[1] Strathclyde AI and Games Research Group, University of Strathclyde. Glasgow, UK. (Corresponding Author : Luke Dicken email luke@cis.strath.ac.uk)

overcome both of these obstacles to Poker research by providing a casino-like environment from which complete data can be extracted, enabling researchers to get a much more accurate picture of the decisions players make and the situations that lead to these decisions. In addition, SPREE provides an experimental platform on which analysis can be performed to properly compare different agent implementations by offering proper experimental practices to be followed, such as replicating all conditions including card order and opponent play-style between tests of two separate agents. Developing agents is a relatively simple task, with a complete framework for this being available in the Java language, and the communications protocol to the SPREE server being available for development in other languages.

1.2 Paper Outline

The remainder of this paper is structured as follows. In Section 2, we will give an outline of the principal components of the SPREE system, namely the server implementation, the current GUI client for human players, the administration interface and the framework for developing agents. In Section 3 we will discuss the uses of SPREE with specific reference to two use cases, one in which SPREE is used to gather information on human Poker playing patterns and another in which SPREE is used as a tool to comparatively evaluate two different Poker agents. Finally, in Section 4 we will summarise the contribution that SPREE makes to the field.

2 SPREE OVERVIEW

The SPREE system is broken into a number of distinct components, each of which will be outlined below. An overview of the interactions of each component is presented in Fig. 1. All components interact by TCP/IP except where otherwise noted.

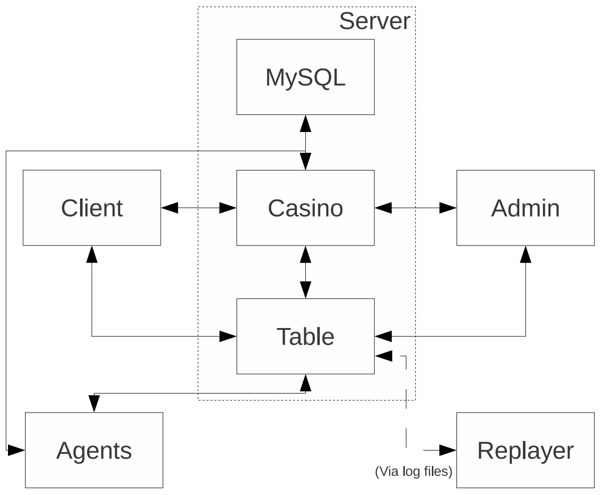


Figure 1. Representation of the interactions of the different components of the SPREE system.

2.1 Server

The core of SPREE is the server, which is used to control the flow of the poker game, including dealing cards and handling the betting options available to each player as play progresses. At its heart, the SPREE server relies on basic technologies, it is implemented in Java and utilises standard TCP/IP communications for negotiating with players. It also requires access to a MySQL server which holds configuration settings and details relating to user accounts.

2.1.1 Casino

The casino component of the server is responsible for controlling what games are on offer and logging players in. This is the component that users initially connect to in order both to authenticate themselves (or register as a new player) and to retrieve a list of Poker tables currently active on the server, the configurations of these tables and their location. This information is stored in a MySQL database (meaning that it is persistent between server restarts). The casino is also responsible for managing a player's "bankroll", or the amount of virtual money they have associated with their account. This amount can be altered by administrators of the SPREE server (more on this below). When a player connects to the casino their current bankroll is passed to the client, when they sit at a table they must choose an amount to take to the table which is deducted from their bankroll. They may then gamble with this amount, and when they leave the table the database is updated, adding the amount they have remaining from this back into their total bankroll.

From a user's perspective, the principal use of the casino is to retrieve the list of available tables. The user can then use this list to connect to the table and communicate with it directly.

2.1.2 Table

Each table runs as a self-contained process initiated by the casino. The table component runs a single table of Poker with a specific rules configuration. The basic game rules are set out using an XML-based game definition, in a format similar to that used by the AAAI Annual Computer Poker Competition[1]. The current version of SPREE can only handle the Texas Hold 'Em variant of the game due primarily to limitations imposed by the client, but by using a formalism such as this for the server, SPREE is quite extensible to other variants of Poker. The game definition used for SPREE's Limit version of Texas Hold 'Em is shown below.

```
<GameDef>
    <Description>Texas Hold’Em</Description>
    <Rounds>4</Rounds>
    <MinPlayers>2</MinPlayers>
    <MaxPlayers>10</MaxPlayers>
    <MinBet>2</MinBet>
    <SmallBlind>1</SmallBlind>
    <BlindStructure>1|2</BlindStructure>
    <PrivateCards>2|0|0|0</PrivateCards>
    <PublicCards>0|3|1|1</PublicCards>
    <BetsPerRound>3|4|4|4</BetsPerRound>
    <BetStructure>1|1|2|2</BetStructure>
</Gamedef>
```

In this example, the Description element gives a text description of the game, which is used by the server to advertise what type of game this is. The Rounds element indicates how many rounds of both dealing and betting occur in the game. MinPlayers and MaxPlayers respectively define the minimum players required to make a game and maximum. MinBet defines the base bet amount for the table. The SmallBlind is traditionally set at half of this bet amount.

The BlindStructure element reflects the positions at the table and the amounts that they must pay prior to the game begining, relative to the SmallBlind amount, so in this case there are two blinds required, with the first being equivalent to one SmallBlind and the other being equivalent to two SmallBlinds. The manner in which cards are dealt to players is determined by the PrivateCards element, which in the example shows that two cards are dealt to each player in the initial round and none for the remaining rounds. PublicCards reflects the manner in which community cards are dealt, and for Texas Hold 'Em this follows the pattern of none in the first round, three in the second and then one each for the third and fourth rounds. BetsPerRound sets a cap on the number of times a bet or raise is allowed, in Limit play this is typically capped at three raises per round, and as the initial bet would not be counted and the blinds are not counted, this is achieved by specifying a three bet cap in the first round and four in the others. BetStructure allows for a common mechanic in which the minimum bet amount increases in the later rounds of the game, this usually doubles, and as the example shows, the way of signalling this in the game definition is by stipulating the bet amount in each round with reference to the MinBet value. In rounds one and two this is equivalent to 1*MinBet, increasing in rounds three and four to 2*MinBet.

When the table is started it does nothing but wait for connections. A game does not begin until two players are seated at the table. At this point, the server begins to deal the cards to the players according to the game definition. It maintains the current state of the hand and contacts each client using a TCP/IP message called a `gameMessage`. There are three possible types of `gameMessage`, a `stateMessage` which details the current state of the game, an `optionsMessage` that offers the client the opportunity to choose one of the available options and an `actionMessage` that is used as a partial update to indicate the action taken by a specific player. This means that at the begining of each round of betting a `stateMessage` is generated which contains the state of the game after the server has dealt cards and otherwise processed the necessary steps between rounds of betting. The `stateMessage` contains information about each player, but replaces cards that the client being contacted is not able to view (such as other players' hole cards) with 'x' to show that information is being hidden. Note that below we will describe an administrative view of this data which is based on this same `stateMessage` data without such obfuscation.

2.1.3 *Exported Data*

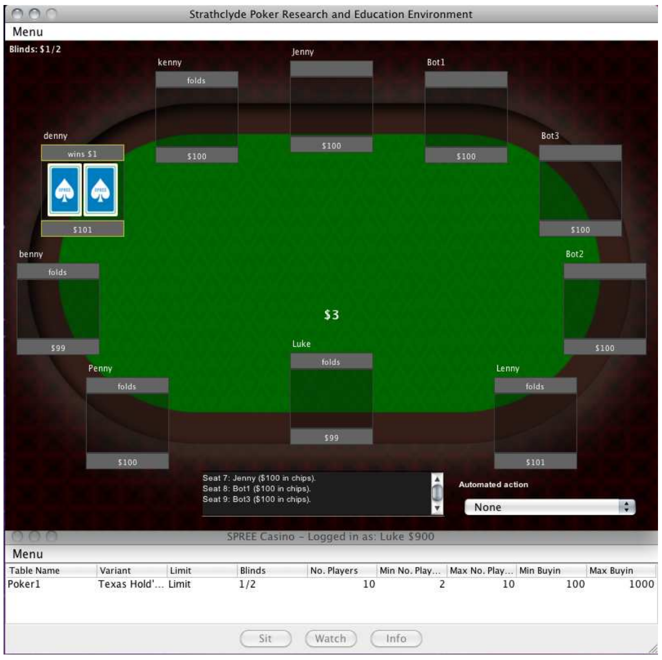
The data that the table records is stored in two formats. The first is a "Recording", which contains the `stateMessage` transmissions that the server has generated during the game. These are used by the replayer component (described below) to step through the game move by move and visualise exactly what happened during that specific game.

The second style of exported data which the server generates is a "Hand History" file, which is a descriptive account of what has occurred during the game. This style of data is used by existing tools such as Poker Tracker[2]. The aim of specifically exporting this data is to allow games played within SPREE to be analysed in the same manner as other games that have been played on other systems, and to maximise interoperability between SPREE and other tools in use both by players and by researchers.

Additionally, note that this export system is designed to be highly modular, and implementations that output the data in a different format can be easily developed if it is found necessary to extract specific data that cannot be parsed from the current structure.

2.2 Client

Our client implementation allows a user to connect to a SPREE Casino and either authenticate to an existing player account or create a new one. From there, a listing of the active tables is retrieved and displayed to the user. An example of this is shown in the lower portion of Fig. 2, where it can be seen that the casino has only one table currently active. Details about the tables are made available to the user to aid them in choosing an appropriate place to play, in much the same way as would be experienced in a real online casino. Specifically, the client displays the name of the table, the particular variant of Poker being played, whether the table is playing Limit or No-Limit, the size of the initial amount wagered, the current number of players at the table, the minimum and maximum players allowed and the minimum and maximum buy-in amounts allowed. The user then has the option to "sit" or "watch" the table. Sitting means that the user intends to play at this table, whilst watching is simply observing the game currently in progress without taking an active part in the game. Additionally, users may request more information about the table.

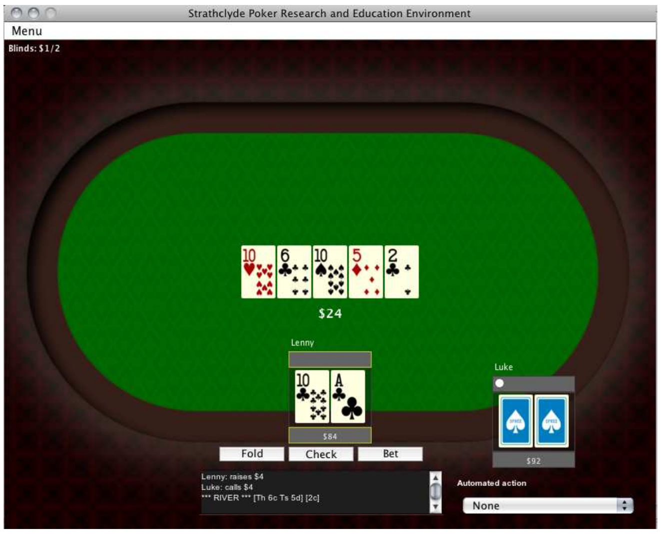


```
Seat 7: Jenny ($100 in chips).
Seat 8: Bot1 ($100 in chips).
Seat 9: Bot3 ($100 in chips).
```

Table Name	Variant	Limit	Blinds	No. Players	Min No. Play...	Max No. Play...	Min Buyin	Max Buyin
Poker1	Texas Hold'...	Limit	1/2	10	2	10	100	1000

Figure 2. Screenshot of SPREE Client. Luke is playing against 9 opponents, Denny won the previous hand.

When a table is selected by the player, the table view portion of the GUI opens, as shown in the upper portion of Fig. 2. This is influenced heavily by the look and feel of a variety of graphical Poker systems, and is relatively representative of a traditional table game of Poker. Fig. 3 shows a game in progress, in which the community cards can be see. Also in this image, the options available to the player are clearly visible. Lenny has the option to fold check or bet. It is also possible to use the GUI to "queue up" an action in advance, meaning that they can make decisions when they first see their cards and decide what they will do (such as fold) before it is their turn. This is a common feature of many casino interfaces as it allows the player to focus their attention elsewhere rather than forcing them to wait patiently for their turn.



```
Lenny: raises $4
Luke: calls $4
*** RIVER *** [Th 6c Ts 5d] [2c]
```

Figure 3. Screenshot of a Poker game in progress. Lenny is playing against Luke. The game has reached its final phase, with Lenny to bet next.

In all respects, the design of the client tried to closely emulate the manner in which online casinos present the game to players, on the assumption that since these casinos succeed in getting players to play in their free time, the features that they offer to players must be of use. Additionally, it is hoped that those already familiar with playing in a commercial environment will find the SPREE implementation roughly equivalent to the experience they are used to.

2.3 Administration Interface

The SPREE administration interface is designed to allow users of the SPREE system who are flagged as administrators to control and configure an active SPREE server. This gives them the opportunity to alter the set of tables currently used by the server by deleting existing ones or adding new ones, and to specify the parameters and game type of these tables as they are added. Administrators are also able to edit user accounts, both by changing usernames and passwords as well as altering the bankroll of a user. It is possible for administrators to make other users into administrators or remove a user's administrator status. It is possible to watch a game in progress at a table from within the administration interface, which provides a real-time perfect information account of the table, including each player's hole cards. This is obviously open to abuse by players who are also administrators, therefore we have enforced that a player who is playing at a table cannot also view that table using the administration interface.

As a separate part of the administration tools, we have also created a "replayer" system which is able to parse the generated hand history files and play back the game step by step, allowing for a view similar to that seen in the administration interface itself, but after the fact rather than in real time. This allows the administrators to inspect the actions that occur within a game, either to look for irregularities in play or perhaps to analyse an agent's behaviour under certain circumstances more closely. This tool operates offline based entirely from the file generated, with no requirement of access to a SPREE server.

2.4 Poker Agent Framework

The SPREE Agent Framework is derived from the codebase of the client component, with the interactive components removed. Like the client, the Agent Framework is built to maintain an internal model of the current Poker game in progress, which is kept current by the receipt of update messages from the server.

When a decision is required of an agent, the `makeMove` routine of the framework is triggered and is passed the possible options that the agent can choose between. This allows the agent developer to insert whatever algorithm they choose into their implementation and send their choice back to the server through the `PokerTableController` library provided. This provides a complete solution for researchers to create Poker playing agents and deploy them by removing any necessity for development on components not directly related to the AI research.

3 POSSIBLE APPLICATIONS

3.1 Data Gathering

As the SPREE system allows researchers to gather complete information about games of Poker that have been played, one of the primary uses of the system is to expand the amount of data available for machine learning to take advantage of, and to do so using these complete hand histories rather than the partial information variants. In order to achieve this, human players need to play Poker within the SPREE environment. Despite the information being complete, a large amount of data will still need to be collected to be of use to the kind of AI techniques that could benefit from it.

The exact method by which this data will be gathered will vary from project to project, however it is important to note when generating such data by playing games involving humans that the perceived value of the virtual money being wagered must be taken into consideration. It is a common phenomenon that when wagering something that has no intrinsic worth to the player (referred to as "play money"), they will typically behave in a quite different manner to players who are wagering something of minor value, such a "low stakes" game, who play as different again from players wagering something they consider to be of significant value, such as in a "high stakes game". Because of this, it may be necessary to somehow offer incentives during the data gathering such that the players involved take it seriously. In any event, when using the SPREE system for data gathering, this issue must be considered as it could potentially have a significant impact on the worth of the data. In the case of machine learning, not accounting for this variance in play style could lead to an agent learning specifically how to beat play money humans, which would be of limited use in a wider context. It is also worth observing that the SPREE environment would be an ideal tool in order to analyse and attempt to quantify exactly what effect these differing stakes have on players' behaviour.

3.2 Agent Evaluation

Because the SPREE system allows for specific scenarios to be established, it is a very good test bed for comparing two different Poker playing agents. Typically, the strength of a Poker player is rated as "Big Bets per 100 Hands", which analyses the player's winnings (in terms of the minimum stake at the table) over time. A standard method of comparison is to use this (or some other "bottom line" metric) on the assumption that over time, two agents will have encountered a similar range of circumstances. As was noted earlier, there are 7.41×10^{39} possible ways of dealing out the cards required for a game of Poker. While a number of these configurations are functionally equivalent (as suits are not ranked in Poker) this still makes for an incredibly large number of circumstances to test agents in, in order for them to have seen a comparable set of aggregate circumstances, and therefore making it difficult to minimise the impact of these differing circumstances on the agents' performance.

SPREE allows researchers to take a different tack by allowing for the dealt cards to be established in advance. This allows specific scenarios to be engineered, meaning that the agents can be tested under specific conditions, but it also means that the cards can be dealt identically for two different agents, so they can each face the same opponents, holding the same cards allowing for a true comparison to be made between the agents, rather than a statistical analysis that attempts to factor out the inherent variance in the results. This feature may also be of use when developing agents, as it allows for a specific scenario in which the agent performs poorly to be identified, and then those cases to be specifically retested as the agent is refined, allowing for a much more targeted analysis of whether these changes are improving the performance of the agent.

4 DISCUSSION AND SUMMARY

At this time we feel that SPREE is quite a mature project, however it is far from complete and there are a range of features that would be of benefit if they were incorporated. Currently, all games that are handled are of a "cash" variety, in which players sit, play for virtual money and may at any time leave the game, retaining the virtual money they had remaining at that point in the game. This is different from the "tournament" style of play, in which a player wagers their money to enter the tournament, receives a number of chips as part of this "buy in" process and then plays until eliminated with no opportunity to leave without forfeiting the game and their wager. It is currently possible to play pseudo-tournaments by placing meta-rules on the players external to the environment itself to achieve the same effect, but this would be significantly more elegant if it was possible within SPREE, and would open up further avenues for analysis since a player's betting style in this format differs significantly to cash games due to the emphasis being more on eliminating players from the tournament rather than ensuring a steady return on wagers.

One of the most obvious features that is currently not implemented within SPREE is a chat system to allow players to communicate with each other. Although this is very common on commercial poker systems, it has deliberately been omitted from the early work developing SPREE as we felt that the presence of a chat system was likely to result in an environment in which detecting agents was trivial as these would be the players not communicating. We wanted the emphasis in designing agents to remain firmly on the algorithmic side, rather than allow the introduction of a Turing Test via a chat system. This may be introduced at a later date, but would probably be optional on a table by table basis to allow agent-based experiments to be conducted, for example one in which humans attempt to detect bots based purely on play style.

Poker research is a very active topic, with many questions outstanding in a variety of areas from developing stronger AI routines to creating agents that can pass themselves off as human players. We have developed SPREE as a tool for our own use to aid us in our work towards answering these questions, and in this paper we introduce it to the wider community in the hopes that will be of similar use to others.

REFERENCES

[1] 'AAAI Annual Computer Poker Competition', *http://www.computerpokercompetition.org/*.

[2] 'Poker Tracker', *http://www.pokertracker.com/*.

[3] Darse Billings, *Algorithms and Assessment in Computer Poker*, Ph.D. dissertation, University of Alberta, 2006.

[4] B Bouzy, 'Computer Go: An AI oriented survey', *Artificial Intelligence*, **132**(1), 39–103, (October 2001).

[5] Guy Van Den Broeck, Kurt Driessens, and Jan Ramon, 'Monte-Carlo Tree Search in Poker using Expected Reward Distributions', *ACML*, 1–15, (2009).

[6] Richard G Carter, 'An Investigation into Tournament Poker Strategy using Evolutionary Algorithms', *Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Games*, (2007).

[7] Fenghsiung Hsu, *Behind Deep Blue: Building the Computer that Defeated the World Chess Champion*, Princeton University Press, 2002.