BAYESIAN OPPONENT MODELING IN ADVERSARIAL GAME ENVIRONMENTS

Roderick James Samuel BAKER

submitted for the degree
of Doctor of Philosophy

Department of Computing

University of Bradford

2010

Abstract

Keywords: Opponent Modeling, Imperfect Information Games, Poker, Neuroevolution, Evolutionary Algorithm, Coevolution.

This thesis investigates the use of Bayesian analysis upon an opponent's behaviour in order to determine the desired goals or strategy used by a given adversary. A terrain analysis approach utilising the A* algorithm is investigated, where a probability distribution between discrete behaviours of an opponent relative to a set of possible goals is generated. The Bayesian analysis of agent behaviour accurately determines the intended goal of an opponent agent, even when the opponent's actions are altered randomly. The environment of Poker is introduced and abstracted for ease of analysis. Bayes' theorem is used to generate an effective opponent model, categorizing behaviour according to its similarity with known styles of opponent. The accuracy of Bayes' rule yields a notable improvement in the performance of an agent once an opponent's style is understood. A hybrid of the Bayesian style predictor and a neuroevolutionary approach is shown to lead to effective dynamic play, in comparison to agents that do not use an opponent model. The use of recurrence in evolved networks is also shown to improve the performance and generalizability of an agent in a multiplayer environment. These strategies are then employed in the full-scale environment of Texas Hold'em, where a betting round-based approach proves useful in determining and counteracting an opponent's play. It is shown that the use of opponent models, with the adaptive benefits of neuroevolution aid the performance of an agent, even when the behaviour of an opponent does not necessarily fit within the strict definitions of opponent 'style'.

Acknowledgements

I would like to thank my research supervisors Prof. Peter Cowling and Dr. Ping Jiang, whose support and guidance (and criticism!) have been vital to getting me through my PhD. I am grateful to Peter in particular, who saw enough potential in me as an undergraduate to give me the opportunity to further my education. Furthermore, I wish to thank all the past members of the now-defunct MOSAIC research group, whose discussions and friendship helped mould me into the researcher I am today.

Finally, a very special thanks to Jennifer Greene, whose boundless faith and support have helped me immeasurably over the past eight years.

This work is funded by an Engineering and Physical Sciences Research Council (EPSRC) studentship.

For Jen.

Contents

Abstract ii

Acknowledgements iii

Contents v

List of Figures x

List of Tables xvi

Chapter 1 : Introduction 1

1.1 Contributions of this Thesis 5

1.2 Authored Academic Papers 6

1.3 Co-Authored Academic Papers 8

1.4 Thesis Outline 9

Chapter 2 : Literature Review 11

2.1 AI and Games 11

2.1.1 Game Tree Search 11

2.1.2 Beyond Minimax 14

2.1.3 NeuroEvolution 15

2.1.4 Perfect and Imperfect Information 20

2.1.5 Pathfinding 23

2.2 Opponent Modeling 24

2.2.1 Human Behaviour 24

2.2.2 Opponent Modeling in Game Playing 26

2.3 Poker 29

2.3.1 Early Work 30

2.3.2 Game Theoretic Approaches 32

2.3.3 Opponent Modeling and Poker 37

2.4 Summary 39

Chapter 3 : Agent Path Prediction 41

3.1 Bayes' Rule 41

3.2 Agent Path Prediction 42

3.2.1 GridWorld 43

3.2.2 Halmoids 44

3.3 Terrain Analysis 46

3.3.1 A* Analysis 47

3.3.2 Potential for Probabilistic Error 48

3.4 Coping with Random Behaviour 53

3.5 Summary 54

Chapter 4 : Strategy Recognition in Simplified Poker 57

4.1 One Card Poker 57

4.1.1 Game Rules 58

4.2 The AI Players 59

4.2.1 Distinct Style Players 59

4.2.2 Implementing the Distinct Styles 60

4.2.3 Design of the Anti-Players 61

4.2.4 Modeling the Opponent using Bayes' Rule 62

4.2.5 Modeling Correct Responses for Differing Styles 67

4.2.6 The Simulation Player 69
4.3 Results and Analysis 70
4.3.1 Bayesian Analysis vs. Simulation Player 70
4.3.2 Performance against Static Styles 74
4.4 Summary 76
Chapter 5 : Evolutionary Approaches to Simplified Poker 77
5.1 Evolving Genetically-Coded 'Anti'-Players 78
5.1.1 Evolving against all Types of Opponent 83
5.2 Evolving an ANN-Controlled Opponent 86
5.2.1 SharpNEAT 87
5.2.2 Evolving NEAT to Play Poker 88
5.2.3 Augmented Evolution with Bayes' Rule 95
5.3 Increasing the Number of Players 97
5.3.1 Recurrency Example 98
5.4 Dynamic opponents 107
5.4.1 Bluffing 108
5.4.2 Play Style Adaptation 110
5.5 Discussion 111
5.6 Summary 113
Chapter 6 : Adaptation to Nonspecific Opponent Styles 114
6.1 Evolutionary Opponents 114
6.2 Coevolutionary Challenges 117

6.3 Chromosome Representation 118

6.4 Coevolution 119

6.5 Summary 128

Chapter 7 : Texas Hold'em 130

7.1 Texas Hold'em Poker 130

7.1.1 A Typical Poker Hand 131

7.1.2 Tournament Format 133

7.2 The AI Players 133

7.2.1 Evolving an ANN-Controlled Opponent 134

7.3 Experimental Results 137

7.4 Bayesian Analysis 142

The Bayesian Observation Window 144

7.5 Observing the Evolved Performance 148

7.6 Round-Based Modeling 151

7.7 Response to Bluffing 161

7.8 Dynamic opponents 164

7.9 Multi-Player Environment 165

7.10 Discussion 171

7.11 Summary 172

Chapter 8 : Summary and Conclusions 174

8.1 Future Work 179

8.1.1 A Preference-Based Modeling Approach to Real-Time Game Environments 179

8.1.2 A Bluff-Aware Approach for Modeling an Opponent's Potential for Deceptive Play 181

8.1.3 Look-Ahead Prediction of Potential Strategy Adaptation 182

8.1.4 Modeling Strategy Recognition Through Idiosyncratic Behaviour Capture . 183

Glossary 185

Appendix A 187

Bibliography 195

List of Figures

Figure 3.1: A simple ship-blocking terrain 45

Figure 3.2: Representation of Move counter in relation to related motion direction ... 48

Figure 3.3: Bayesian analysis of an opponent ship heading towards Target 3 (right-hand treasure chest) 51

Figure 3.4: Bayesian analysis of an opponent ship heading towards Target 2 (central treasure chest) 52

Figure 3.5: Performance of the Bayes' Rule -Controlled (Green/upper) ship 52

Figure 3.6: Interception success of a Bayes' Rule-controlled ship against an opponent applying varying levels of randomness to its actions 53

Figure 4.1: Layout of a General Player 60

Figure 4.2 : Graph displaying how player type probabilities alter given opponent actions. In this case, the analysis player converges to the conclusion that the opponent is a Tight Passive player 66

Figure 4.3: Graph displaying combinations of opponent styles against the percentage of tournaments won by each player 70

Figure 4.4: Graph displaying the tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LA/LA opponents 72

Figure 4.5: Graph displaying the tournament success of different values of α and β when playing a four-player tournament against TP/TP/TP opponents 73

Figure 4.6: Graph displaying the tournament success of different values of α and β when playing a four-player tournament against LA/LA/TP opponents. 73

Figure 4.7: Graph displaying the tournament success of different values of α and β when playing a four-player tournament against LA/LA/LP opponents. 74

Figure 5.1: Design of the Chromosome 78

Figure 5.2: Evolution of a GA player against a single Loose Aggressive Opponent 79

Figure 5.3: Evolution of a GA player against a single Loose Passive Opponent 80

Figure 5.4: Evolution of a GA player against a single Tight Aggressive Opponent 80

Figure 5.5: Evolution of a GA player against a single Tight Passive Opponent 81

Figure 5.6: Evolution of a GA player against all types of opponent 84

Figure 5.7: Evolution of a GA player using the Bayesian Opponent Model against all opponent styles 85

Figure 5.8: Design of player network. 88

Figure 5.9: Evolution of an ANN against a simple Loose Aggressive Player. 91

Figure 5.10: Evolution of an ANN against a simple Loose Passive Player 91

Figure 5.11: Evolution of an ANN against a simple Tight Aggressive Player 92

Figure 5.12: Network Structure of the evolved best network from The evolution shown in Figure 5.11 93

Figure 5.13: Evolution of an ANN against a simple Tight Passive Player 93

Figure 5.14: Evolution of an ANN against all types of opponent 95

Figure 5.15: Design of proposed player 96

Figure 5.16: Evolution of a Bayesian Model-augmented ANN against all opponent styles.....97
Figure 5.17: Design of a style-predictive ANN 99
Figure 5.18: Evolution of a 14-input ANN analysing opponent style.....99
Figure 5.19: Evolution of a 2-input recurrent ANN analysing opponent style 101
Figure 5.20: Evolution of a three-input recurrent network against all four types of opponent in a four-player scenario 102
Figure 5.21: Evolution of a Bayesian-augmented ANN against all types of opponent in a four-player scenario.....103
Figure 5.22: Network structure of an evolved network from Figure 5.21.....104
Figure 5.23: Tournament performance of the best evolved genome against all combinations of 3 opponents over 100 tournaments – The player was evolved against only the four highlighted combinations of players.....105
Figure 5.24: Tournament performance of ablated configurations of the 'best' evolved genome over 100 tournaments 106
Figure 5.25: Tournament performance of the 'best' evolved genome against partially randomized opponents over 100 tournaments.....108
Figure 5.26: Bayesian analysis of a TP player with varying bluff probabilities 110
Figure 5.27: Tournament performance of the 'best' evolved genome against dynamically styled opponents over 100 tournaments 111
Figure 6.1: Evolution of a probability-controlled GA player against the best evolved ANN 115

Figure 6.2: Evolution of a Bayes' rule Utilising ANN against the GA player 117

Figure 6.3: Pseudocode of the Shared Fitness Function.....120

Figure 6.4: Coevolution of two populations using a 'threshold' size of 10 122

Figure 6.5: Performance of a static LA player against the Best evolved solutions from each generation. 123

Figure 6.6: Performance of a static LP player against the Best evolved solutions from each generation. 124

Figure 6.7: Performance of a static TA player against the Best evolved solutions from each generation. 124

Figure 6.8: Performance of a static TP player against the Best evolved solutions from each generation. 125

Figure 6.9: Performance of the Bayesian ANN Player against the Best evolved solutions from population 1. 126

Figure 6.10: Performance of the Bayesian ANN Player against the Best evolved solutions from population 2.....126

Figure 7.1: Design of player network 135

Figure 7.2: Evolution of an ANN against a Loose Aggressive Player.....139

Figure 7.3: Evolution of an ANN against a Loose Passive Player 139

Figure 7.4: Evolution of an ANN against a Tight Aggressive Player 140

Figure 7.5: Evolution of an ANN against a Tight Passive Player.....140

Figure 7.6: Evolution of an ANN against all four player styles.....142

Figure 7.7: Design of proposed player 143

Figure 7.8: Probability convergence of the Bayesian model given a set of LA-type actions. 145
Figure 7.9: Comparison of probability convergence using standard Bayes' rule (top), and windowed analysis (bottom) The opponent changes style from Tight Passive to Loose Aggressive at action number 7. 146
Figure 7.10: Evolution of a Bayesian Model-augmented ANN against all opponent styles..... 147
Figure 7.11: Performance of the 'best' evolved ANN against a Loose Aggressive Player. 148
Figure 7.12: Performance of the 'best' evolved ANN against a Loose Passive Player.. 149
Figure 7.13: Performance of the 'best' evolved ANN against a Tight Aggressive Player. 149
Figure 7.14: Performance of the 'best' evolved ANN against a Tight Passive Player. .. 150
Figure 7.15: Analysis of a Loose Passive player using a single-round Bayesian approach 153
Figure 7.16: Bayesian Multi-Model Analysis Vs. A Loose Aggressive Player 155
Figure 7.17: Bayesian Multi-Model Analysis Vs. A Loose Passive Player..... 155
Figure 7.18: Multi Bayesian Model Analysis Vs. A Tight Aggressive Player..... 156
Figure 7.19: Multi Bayesian Model Analysis Vs. A Tight Passive Player 156
Figure 7.20: Evolution of a multi-round Bayesian player. 157
Figure 7.21: Performance of the 'best' evolved ANNs using each strategy against a Loose Aggressive Player. 158

Figure 7.22: Performance of the ‘best’ evolved ANNs using each strategy against a Loose Passive Player.....159

Figure 7.23: Performance of the ‘best’ evolved ANNs using each strategy against a Tight Aggressive Player. 159

Figure 7.24: Performance of the ‘best’ evolved ANNs using each strategy against a Tight Passive Player.....160

Figure 7.25: Performance of an evolved ANN versus a Tight Passive player with a varied bluffing level.....162

Figure 7.26: Performance of an evolved Multi-Round Bayesian ANN versus a Tight Passive player with a varied bluffing level. 163

Figure 7.27: Performance of evolved agent against a dynamic opponent (with representation of player’s belief of opponent style).....164

Figure 7.28: Evolution of a recurrent network (without opponent model) against all four types of opponent in a four-player scenario.....167

Figure 7.29: Evolution of a recurrent network (with opponent model) against all four types of opponent in a four-player scenario. 168

Figure 7.30: Design of a 30-input ANN.....169

Figure 7.31: Evolution of a non- recurrent, 30-input network (with opponent model) against all four types of opponent in a four-player scenario. 169

Figure 7.32: Tournament performance of the best evolved genome against all combinations of 3 opponents over 300 hands – The player was evolved against only the four combinations of players highlighted in black. 171

List of Tables

Table 3.1: Probability distribution over all possible actions per available opponent targets from Figure 3.2. 50

Table 4.1: α and β values for each style of deterministic player. 61

Table 4.2: α and β values for each style of Anti-Player. 62

Table 4.3: Action probabilities for each opponent style. 65

Table 7.1: Action probabilities for each opponent style. 144

Table 7.2: Action probabilities for each opponent style per betting round. 152

List of Algorithms

Algorithm 4.1: Pseudocode of the Bayes' Theorem predictor 64

Algorithm 4.2: Pseudocode for the Analysis Anti-Player to determine the opponents' style and choose which tactic to employ 69

Chapter 1: Introduction

Inference is the derivation of a conclusion based solely upon the information presented to the being in question; it goes unnoticed, but the human brain is constantly applying a heuristic evaluation to everyday situations. For example, if a ball is falling and we aim to catch it, then we infer from the direction and speed it is travelling how fast to move in order to do so. Likewise in a situation when crossing a road, if a vehicle is moving towards us at great speed, we infer whether or not to cross based on what we know about our own speed capabilities.

Humans cannot normally explicitly explain the fine points of exactly what we are thinking at the time of inference, just that we *knew* that the actions we performed were correct at that time. Some would argue that our mind's instant calculation and force of action would support the case of a *materialist* viewpoint, where all phenomena are predetermined by the mechanics of the brain rather than a higher-level free will, but this is beyond the scope of this thesis. The interested reader may refer to "*L'homme Machine*" (Man a Machine) by Julien Offray de la Mettrie (1748) for a philosophical discussion of this viewpoint. Our interest is somewhat analogous to this observation, as Artificial Intelligence (AI) is the process of understanding the mechanics of human inference, and thus designing computer methods to simulate them, or (at the least) to solve problems which otherwise would require ingenuity of the type shown by humans. Artificial Intelligence can be defined as the study and design of agents which display intelligence, more specifically *human* intelligence. It is a

science which strives to further our understanding our own and other animal intelligence (Callan, 2003) (Finlay & Dix, 1997) (Negnevitsky, 2005).

Many people, upon the first mention of AI, will probably relate their experiences to modern events and entertainment. One of the most high-profile events has been that of Garry Kasparov and his famous defeat to (and subsequent controversy surrounding) Deep Blue in the game of Chess (Campbell et al., 2002). Furthermore, the recent solving of the game of Checkers is a landmark achievement in the field of AI (Schaeffer et al., 2007). Beyond the scope of such 'traditional' AI however, research into game AI has been a popular topic for several years; Laird's rallying call to research in *commercial* games (Laird & Van Lent, 2001), for example, has become a popular reference point for much research into game AI which, although out-dated in some respects, marks a significant turning point in relation to bringing 'serious' research into a virtual landscape, especially in relation to creating truly adaptive and intelligent game opponents.

Understanding the relevance of an opponent's actions, using them to infer conclusions about the opponent's behaviour is one of the most integral parts of game AI. Games, in general, require some element of prediction and understanding of opponent actions in order to make an effective tactical decision. The capability to predict an opponents' next move is an extremely useful way to defeat an adversary; knowing your enemy is integral to success in all forms of game, whether it is in a virtual video game environment, a game of Poker, predicting changes in stock values, or even

waging war. Game theory (von Neumann & Morgenstern, 1944), and the formal investigation thereof, has probably been best described as "the analysis of rational behaviour under circumstances of strategic interdependence, when an individual's best strategy depends upon what his opponents are likely to do" (Varoufakis, 2001), and is integral to every aspect of modeling an opponent's behaviour. Opponent Modeling is but a small facet of adaptive gameplay, but the understanding of an opponent's strategy or capabilities through inference is a significant starting point for (at least the illusion of) truly intelligent agents.

This thesis aims to apply opponent modelling to a series of environments as a technique to improve an agent's ability to understand and counteract the strategy of a single, or indeed a set of, adversaries. With respect to game playing, Poker in particular, Jonathan Schaeffer's (very laudable) aim is to develop a Poker player than can one day win the World Series of Poker (WSOP); that is not the focus of this thesis (Billings et al., 2001). Our primary focus is to investigate how opponent modelling can improve the performance of an agent in conjunction with other AI techniques such as Pathfinding, Neuroevolution and Genetic evolution, and Poker serves as an ideal environment within which to pursue this goal. The imperfect information environment of Poker offers many challenges relating to the understanding of opponent actions with respect to the underlying strategy behind them, which this thesis aims to address. Furthermore, Carter's work notes that a multi-opponent Poker environment is generally ignored (where Carters work actively ignores opponent models), this is a

major motivation for this work; we not only aim to apply opponent modelling to a poker environment, but also aim to utilise multiple opponent models to improve an agent's play against any number of opponents (Carter & Levine, 2007). Our opponent modelling approach makes heavy use of Bayesian Reasoning, which requires probability values as primary inputs (Negnevitsky, 2005). In our case, these are represented as 'action probabilities' – In the case of expert systems, these values are determined through human judgement and in the case of a game such as poker, human beliefs about the opponent are necessary to determine an accurate strategy to take. The issue with human judgement is that it is inaccurate, and research has shown that humans cannot necessarily elicit probability values consistent with Bayesian rules, or at least make a poor attempt (Burns & Pearl, 1981) (Tversky & Kahneman, 1982). This serves as a motivation for part of this work which applies both a probabilistic terrain analysis to any given game terrain (hence yielding a probabilistically accurate representation of action probabilities) and frequency analysis of opponent actions prior to actually playing the game (meaning that behaviour beliefs are representative of actual play rather than a human belief). In our use of opponent models, we use the NEAT algorithm as a means of evolving agents for game playing, although EANNs have been used for agent evolution (Stanley et al., 2005), the approach used in this thesis focuses upon the use of recurrent ANNs as a tool for improved performance in a multi-player environment.

1.1 Contributions of this Thesis

- An implementation of Bayesian terrain analysis in conjunction with the A* search algorithm to predict the path of an enemy unit in a grid-based strategy game. Chapter 3, (Baker et al., 2008)
- A probabilistic approach to modeling the style of opponent players in a simplified Poker game, using a collection of previously performed actions. Chapter 4, (Baker & Cowling, 2007)
- An investigation into the effect of opponent models upon the quality of evolved solutions in both Genetic and Neural Network evolution. Chapter 5, (Baker et al., 2008)
- Utilisation of the recurrent nature of evolved Artificial Neural Networks to provide a scalable approach to game playing in a multi-agent environment. Chapter 5.3 (Baker et al., 2008)
- An investigation into the stability of the Bayesian opponent model when confronting adversaries that have been Coevolved independent of the standard opponent definitions. Chapter 6
- An adaptation of the opponent model to create a multi-betting round focused player, in order to apply to an implementation of full-scale Texas Hold'em. Chapter 7.

1.2 Authored Academic Papers

The following are references to, and abstracts of, academic papers I have authored (as the main author):

R.J.S. Baker, and P.I. Cowling (2007) – "**Bayesian Opponent Modeling in a Simple Poker Environment**". *IEEE Symposium on Computational Intelligence and Games (CIG 2007)*, Honolulu, USA, pp. 125-131 (Baker & Cowling, 2007)

Abstract -- In this paper, we use a simple Poker game to investigate Bayesian opponent modeling. Opponents are defined in four distinctive styles, and tactics are developed which defeat each of the respective styles. By analysing the past actions of each opponent, and comparing to action related probabilities, the most challenging opponent is identified, and the strategy employed is one that aims to counter that player. The opponent modeling player plays well against non-reactive player styles, and also performs well when compared to a player that knows the exact styles of each opponent in advance.

The content of this paper is covered in depth in Chapter 4.

R.J.S. Baker, P.I. Cowling, T.W.G. Randall, and P. Jiang (2008) – "**Can Opponent Models Aid Poker Player Evolution**?". *IEEE Symposium on Computational Intelligence and Games (CIG 2008)*, Perth, WA, pp. 23-30 (Baker et al., 2008)

Abstract -- We investigate the impact of Bayesian opponent modelling the evolution of a player for a simplified Poker game. Through the evolution of

Artificial Neural Networks using NEAT we create and compare players both utilizing and ignoring Bayesian opponent beliefs. We test the effectiveness of this model against various collections of dynamic and partially randomized opponents and find that using a Bayesian opponent model enhances our AI players even when dealing with a previously unseen collection of players. We further utilize the inherent recurrence of our evolved players in order to recognize the opponent models of multiple players. Through ablative studies upon the inputs of the network, we show that utilization of an opponent model as an evolutionary aid yields significantly stronger players in this case.

The content of this paper is covered in depth in Chapter 5.

R.J.S. Baker, P.I. Cowling, T.W.G. Randall, and P. Jiang – "**Using Bayes' Theorem for Path Prediction**". *9th Informatics Research Workshop for Research Students, University of Bradford, 2008*. (ed. D. Rigas) pp. 101-104 (Baker et al., 2008)

Abstract -- Understanding the intentions of another living creature is an inherent ability, one which humans use with great regularity in daily life. In video games, however, static interactions are generally used instead of ones that dynamically adapt to their opponents. This paper discusses the potential of opponent models and their application to predicting player goals in video games. The work presented uses a grid-based interception game to form probabilistic beliefs relating actions to goals by observing path transitions found through the A*

algorithm. We then apply Bayesian inference to determine the opponent's destination given only real-time enemy path information.

The content of this paper is covered in depth in Chapter 3.

1.3 Co-Authored Academic Papers

T.W.G. Randall, P. I. Cowling, R. J. S. Baker (2007). "**Learning Ship Combat Strategies in the Commercial Video Game DEFCON**". *8th Informatics Research Workshop, University of Bradford.* (ed. D. Rigas) pp. 182-183 (Randall et al., 2007)

Abstract -- In this paper we use a commercial computer game called DEFCON for the creation of player AI that is capable of learning behaviours from the opponents that the agent plays. We then aim to use the knowledge learnt in controlling the other agent for different tasks and in different scenarios.

Role of the Author – The author developed code to circumvent the static AI that was hard-coded into the DEFCON game itself, as well as co-developing a system to load in customised AI player libraries into the game. Furthermore, the author contributed some experimental ideas to the design of agent behaviours for the learning algorithm to mimic.

Randall T.W.G., Cowling P.I., Baker R.J.S. and Jiang P. (2009): "**Using Neural Networks for Strategy Selection in Real-Time Strategy Games**", *Proc. AISB Symposium on AI & Games*, Edinburgh, UK (Randall et al., 2009)

Abstract -- Video games continue to grow in importance as a platform for Artificial Intelligence (AI) research since they offer a rich virtual environment without the noise present in the real world. In this paper, a simulated ship combat game is used as an environment for evolving neural network controlled ship combat strategies. Domain knowledge is used as input to the Artificial Neural Networks (ANNs) through scripts that run in parallel and feed their decisions to the ANNs. The ANNs then interpret these scripts and decide what strategy to perform. The results are compared to ANNs that have no such knowledge and tested to see how well the ANNs generalise.

Role of the Author – The author co-developed DEFSIM, a ship-combat game that emulates the ship behaviours of the commercial game DEFCON for fast testing of the evolved ANN player, which is used in this paper. The author also designed some of the agent-testing scenarios for use in the paper, as well as contributing the visual examples of the aforementioned testing scenarios.

1.4 Thesis Outline

The thesis is structured as follows: Chapter 2 discusses some of the literature related to this research, namely opponent modeling and game-playing with a greater focus upon the game of Poker which shall be a significant focus of this work. Chapter 3 introduces our use of Bayesian probabilities to determine the goals of an opponent in a simple grid based game, Halmoids, combining an opponent modeling approach with A* search to form a probabilistic terrain analysis. Chapter 4 investigates the application

of our Bayesian model to a simplified version of the game of Poker, and its utility in selecting a suitable response strategy to a set of opponents. Chapter 5 expands upon Chapter 4's work through the introduction of several evolutionary approaches to produce agents for the simple Poker game, and investigates how the developed opponent model affects an evolved agent's capability of coping with different styles of opponent. The model is further utilised in an investigation into how the potentially recurrent nature of evolved neural networks can enable a network to cope with the information of several opponents rather than a one-on-one environment. The stability and robustness of the model and resulting evolved player is further tested in Chapter 6 through a series of Co-Evolutionary opponents of interdependent styles. Chapter 7 takes the progress of the previous chapters, and applies the approaches used to a full-scale implementation of the Texas Hold'em game. This chapter once again uses the recurrent nature of ANNs to play against a collection of several opponents, and also describes the necessary modifications required to enable the Bayesian model to work in a framework with multiple betting rounds. Finally, Chapter 8 discusses some of the conclusions from this thesis, as well as offering directions for further research into the realm of opponent modeling with respect to the environments considered in this document, as well as expansions into further game environments.

Chapter 2: Literature Review

This chapter covers some of the surrounding literature relevant to the research in this thesis. The development of AI in game environments is explored with a discussion of various techniques used in game playing. Our main focus shall be upon opponent modeling, with some discussion of the various techniques applied by other researchers in understanding an opponent's behaviour.

2.1 AI and Games

2.1.1 Game Tree Search

At the very inception of Game Theory, Von Neumann and Morgenstern (von Neumann & Morgenstern, 1944) developed the idea of a minimax search upon the game tree of a zero-sum game (i.e. my loss is your gain). This considers all possible performable actions by searching through the game tree through to its conclusion and maximizing the potential gain. Although an extremely useful approach, there are numerous difficulties when being used for various games. When playing games with excessively complex game trees (Chess and Go being the most notable examples), the computationally expensive nature of searching to the end of the game tree is intractable. In such a complex case, the search can be depth-limited to reduce computation time, but requires a heuristic score for the given nodes at the limit rather than a concrete 'win/lose' or 'gain/loss' analysis. The further performance improvement gained by the depth-first search technique using α-β pruning became an

important factor (Knuth & Moore, 1973), which removes branches of the search where the maximal score of the minimizing player (the opponent at this depth) becomes less than the minimal score of the maximizing player (the player aiming for the best available score at this depth); this situation would imply that both players cannot be playing optimally. This pruning dramatically improves the performance of the minimax search, thus improving the analysis of more complex games. In the case of Chess, Deep Blue utilised the same 'brute force' approach to search and applied heuristic reasoning to the state of the Chessboard at a depth of 12 plies (Campbell et al., 2002). The minimax technique assumes, however, that the opponent is performing to the same standard as the heuristic given to the search. This is limiting in its assumption that the opponent would perform in exactly the same way as the agent given an identical situation, which is not necessarily the case. The approach is further restricted through the limitation of being applicable only to a two-player scenario. Progress is still being made, but only in a direction which harnesses Deep Blue's techniques, transitioning from a (brute force) hardware approach to that of a software one, of which a success over the current Chess world champion Vladimir Kramnik has somewhat proven (Hamilton & Garber, 1997) (BBC News, 2006). The AI research community's love affair with Chess has all but come to an end, as far as to say that no known human exists to beat the current techniques, focus can finally be applied to other, potentially more deserving problems.

Providing a greater focus for minimax search was an aim of Moriarty and Miikkulainen, who utilize ANNs with minimax in the game of Othello (Moriarty & Miikkulainen, 1994). Championship-level artificial Othello players have been investigated (and successful) for some time by Rosenbloom, who used minimax with α-β pruning (and a carefully-constructed evaluation algorithm) to create the Shakespeare-referencing IAGO (Rosenbloom, 1988). Lee and Mahajan expenaded this work with pre-computed tables to increase computational speed, as well as greatly expanding the evaluation function - the resulting player (BILL) consistently beats Rosenbloom's player, whilst using far less computation time (Lee & Mahajan, 1990). Moriarty and Miikkulainen's approach, however, uses genetic algorithms to evolve a neural network that is used to direct the game-tree traversal of minimax search with α-β pruning onto a game of Othello. The approach used is based upon the Symbolic, Adaptive Neuro-Evolution (SANE) coevolutionary system (which is also used in the aforementioned BILL Othello player) (Moriarty & Miikkulainen, 1997). At each search level, the network communicates the most promising path to follow. After testing against a player with the same evaluation function, but no focusing to its search ability, the resulting player was shown to match one of the strongest Othello AIs currently available. The authors comment on the common usage of minimax search in game state-spaces and the inherent limitation of time and storage in relation to attempting to search to a game-end situation. One of the more striking points of this work is that the authors note that searching deeper (without a fitting evaluation function) can actually do more harm to the quality of the resulting play than good. The improvement

of this player was such that the ANN heuristic with a 2-ply search outperformed a full minimax search player at a depth of 4 plies. The efforts following this research (Moriarty & Miikkulainen, 1995) discovered new game-playing strategies against both a random moving opponents and an opponent using α-β pruning. The network subsequently learnt positional and mobility strategies. The mobility strategy is an advanced Othello technique, and is generally used only in tournaments by experts; it is surprising that the agent learnt this technique partly due to the fact that the networks involved learnt the game of Othello with no prior rule input or heuristic.

2.1.2 Beyond Minimax

Chinook, an AI for the game of Checkers developed by Jonathan Schaeffer, has now provided the world with a solution to the game after almost 20 years of intensive computation (Schaeffer et al., 2007). The work finally infers that perfect play by both competitors will result in a draw such that "[it] could play draughts against God and would get a draw" (The Guardian, 2007). Schaeffer's approach uses an iterative search along with retrograde analysis (working backwards from all possible endgames) in order to complete a database of all possible eventualities in the game and therefore solve the game. Conversely, the solution to a game such as Poker is much further from our grasp. In comparison to Schaeffer's brute force approach, Gerald Tesauro's research into Backgammon uses a neural network is used as the evaluation function for the game state. Self-play is utilized along with Temporal Difference learning in order to improve the accuracy of the evaluation (Tesauro, 1995). This is in contrast to

Tesauro's previous work which made use of Backpropagation and a set of expert-game records (Tesauro, 1990). Chellapila and Fogel's work uses a similar approach to Tesauro's, creating the ingeniously named *Blondie24*, utilizing the minimax algorithm in conjunction with an Artificial Neural Network (ANN) as its heuristic scoring function (which receives vector information concerning the game state). The weights of the ANN connections were found using an evolutionary algorithm focusing upon playing against different versions of itself (Chellapilla & Fogel, 1999).

2.1.3 NeuroEvolution

Evolutionary Computation is a branch of AI concerned with the continual improvement of a population of solutions through directed random search in order to adapt to a given problem (De Jong, 1975) (Mitchell, 1996). This 'improvement' is determined by processes which mimic the real-world occurrence of evolution, effectively 'breeding' and 'mutating' chromosomes akin to what would happen in biological processes. Generally, a population *P* of *n* solutions (chromosomes) is randomly generated (typically bit strings, but integer or floating point values may be used) and evaluated with a given fitness function. After evaluation, the following operations are most common:

Selection: Choose chromosomes in the population for reproduction; the greater the fitness score, the greater the chance of reproduction

Crossover: Given a fixed point, the genetic sequence before and after that point is traded between two chromosomes, creating two offspring that comprise information from both parents.

Mutation: Randomly flip bits (in a bit string representation), or alter the value in a random number of genes in a chromosome with a (generally low) probability *p*. Using a real-valued representation, mutation can be performed by adding a value generated through a Gaussian distribution.

An early example of evolutionary computation with respect to games can be found in Axelrod's work into the Iterated Prisoner's Dilemma (IPD) using Holland's genetic Algorithm (Axelrod, 1987) (Holland, 1975). This interest in both evolution and IPD still continues through Mittal and Deb's research into multi-objective evolutionary programming (Mittal & Deb, 2009).

In Coevolution, all members of a population compete against one another, but in competitive coevolution two populations of solutions compete against one another such that an increase in fitness in one population yields a decrease in fitness in another. A good example of this application to games is Rosin and Belew's investigation into competitive CoEvolution, concerning itself improving agents for the game environments of Tic-Tac-Toe and Nim (Rosin & Belew, 1997). In this case, two populations of equal size are generated, within which each solution is measured in fitness against each member of the *opposite* population. Candidate fitness is evaluated using the concept of *shared fitness* which is such that weaker solutions with important,

individual, attributes have a fair chance of survival (regardless of certain weaknesses). The fitness of a solution *s* within Population *A* is determined relating to the number of solutions *s* has defeated from population *B*, with respect to how many other solutions from Population *A* have also defeated that player (i.e. if a generally weak player is able to beat a single opponent that no other player can defeat, it has therefore a fairer chance of surviving due to its individuality).

Artificial Neural Networks are parallel computational models inspired by the structure and function of biological nervous systems. The earliest work is generally considered to be McCulloch and Pitts' neuron model (McCulloch & Pitts, 1943), which took a number of inputs, and 'fires' through a single output (0 or 1) depending upon if the inputs surpass a given activation threshold. Rosenblatt further developed this work creating a network using the 'Perceptron', which outputs 1 or -1 depending upon the weighted, linear combination of inputs (Rosenblatt, 1959). Beyond the 1970s, research into ANNs waned due to disinterest until it was later found that the limitations of the perceptron (primarily an inability to represent linearly inseparable functions) could be overcome through the use of multiple hidden layers of neurons to develop an approach to solving complex classification problems (Picton, 2000). This approach has been widely adopted in the realm of Computational Intelligence, with applications from fingerprint recognition (Leung et al., 1990), to explosive detection in airline baggage (Shea & Liu, 1990).

ANNs are generally applied to classification and data mining tasks, and as such the number of outputs is limited, and the task of choosing the type and form of network

inputs can prove arduous (Lawrence et al., 1996). ANNs have, however, been used successfully within some video games, such as Colin McRae Rally 2.0, a Rally motorsport simulation game which trained ANNs to control opponent vehicles. This technique can only be applied where a finite number of responses are required. There are pitfalls in this technique, however, as the perceived 'mysterious' nature of ANN's can lead to situations where if a bug in the system is found, or a new output response is required, a new ANN would need to be constructed and trained/evolved.

Neuroevolution (NE) is the application of evolutionary techniques to the process of generating Artificial Neural Networks. Where some NE techniques involve the evolution of only the connection weights of a neural network, Topology and Weight Evolving Neural Networks (TWEANNs) evolve both the weights of the network as well as the *topology*. An early NE technique is the aforementioned SANE, which evolves populations of neurons instead of populations of networks. These are used as the hidden layer of a (fully-connected) network, within which their performance is evaluated. Each of the evolved solutions receives the average fitness of all networks within which they appeared. Neuroevolutionary approaches into investigating games have been primarily interested in the evolution of an agent directly responsible for game playing behaviour. The board game of Go was of particular interest, where strong players have been evolved for a simpler, scaled down version of the game (Richards et al., 1998). More recent investigations have used the NEAT (NeuroEvolution of Augmenting Topologies) algorithm developed by Ken Stanley, which adapts the topology of an Evolutionary Artificial Neural Network, not just the

weights of connections between neurons (Stanley & Miikkulainen, 2002). An evolutionary algorithm is used, which utilizes historical markings, separating innovations into separate species, and gradually increases the size of the networks involved, determining the structure of the ANN. Each connection gene stores the input and output nodes, as well as the connection weight and, if it is enabled, an 'innovation number' which helps find corresponding genes during genetic crossover. Mutation in NEAT affects nodes, weights and connections, with nodes being added (splitting a connection, disabling one connection and creating two more), or new connections between nodes being created. Crossover involves matching genes between two equal-fitness parents, and using disjoint and excess genes to create an expanded child node. The population is speciated so that innovations are not lost; only new innovations (with the same innovation number) are compared to each other, rather than the entire population. The fitness of a single *genotype* is the determining factor in the existence of the species, which is based upon the averaged fitness of all genotypes within that species. This restriction means that greater populations are at greater risk of more adaptation or replacement.

The adaptation of NEAT into a real-time game environment (Unreal Tournament) initially uses a middleware system, TIELT (Testbed for Integration and Evaluation of Learning Techniques), which aids in the investigation of how an evolutionary Artificial Neural Network may perform in a videogame. The evolution of a game-playing agent is seen to be a very slow process, with limited progress made in simple path finding tasks

(Miikkulainen et al., 2006). ANNs (and by association, Evolutionary ANNs) are more suited to decision-based tasks, and other attempts to introduce Evolutionary ANNs have included strategy selection, and the attempted learning of an opponent's own strategy selection (Randall et al., 2007) (Randall et al., 2009)

The necessity for a fast, real time evolutionary process resulted in rtNEAT, an adapted version of NEAT that evaluates the fitness of evolving agents in a game environment and performs topological adjustment in real-time (Stanley et al., 2005) (D'Silva et al., 2005). The primary evaluation of rtNEAT is within the NERO (NeuroEvolutionary Robotic Operatives) game, which is primarily based upon the training and evaluation of a series of robotic agents to perform a variety of tasks such as approaching an enemy, shooting an enemy and avoiding enemy fire.

2.1.4 Perfect and Imperfect Information

Games can generally be categorized into two major camps; those with, and without perfect information. Perfect information implies that all players can observe all actions in the game, as well as the status of all artefacts that the game consists of (these may be counters, pieces, or cards). Examples of perfect information games include Chess and Checkers in which both players can observe the entire game state on the board. In contrast, an imperfect information game denies players of many facets of the game state in order to play within the realms of uncertainty; Poker is a good example of this, which shall be discussed later in section 2.3 of this chapter. Within the definitions of perfect information is the nature of the stochasticity of the environment; this implies

that unpredictable events take place during the playing of the game. Poker, for example, is a stochastic imperfect information game where cards are shuffled before dealing and opponent cards are hidden during the course of the betting rounds. Backgammon however, is a stochastic perfect information game where the counters are visible to all players, yet the roll of the dice is non-deterministic.

It could be argued that games involving imperfect information have become a much more important focus within the past ten years, shifting from analysis of games where all information can be known, to ones within which the withheld information is the most valuable. Arguably the most notable Imperfect information game is that of the Iterated Prisoner's Dilemma (IPD). The prisoner's dilemma is a nonzero-sum non-cooperative game where two separated suspects (the prisoners) are offered a deal where if one testifies against the other (defect) and the other remains silent (cooperate) then the defector is freed and the co-operator receives a 10 year sentence. If both prisoners defect then both receive a 5 year sentence. However, if both remain silent then only a six-month sentence is bestowed upon each. Each prisoner must choose to stay silent or defect. In the Iterated Prisoner's Dilemma, this game is played repeatedly such that a betrayed player has the opportunity to punish a previously defecting partner (Aumann, 1959). In applying Genetic Algorithms to the game Axelrod found that greedy strategies performed poorly over an extended period of time, whereas altruistic behaviours can eventually prove more profitable (Axelrod, 1987).

Another example of imperfect information on a small, yet still puzzling scale is that of RoShamBo, known generally as the Rock-Paper-Scissors game. This game consists of two players who at the same instance choose one of the three options: *rock*, *paper*, or *scissors*. There are four potential outcomes of each round; Rock beats Scissors, Scissors beats Paper, Paper beats Rock, or both players choose the same option and the outcome is a draw. This is an extremely pure example of a game of imperfect information; the opponent's action is made without any knowledge of your own, and vice versa. This is a situation in which opponent modeling must take precedence, especially in the case of iterated conflicts between the two same opponents, as shall soon be explained, a random strategy can be an optimal one in terms of a game such as this, but deterministic methods are extremely sub-optimal in reference to an opponent which tracks, models, and reacts to past encounters. Darse Billings at the University of Alberta, Canada has run a yearly (sadly, now defunct) RoShamBo competition, in which participants are invited to create artificial players to compete in a tournament of multiple iterations of the game (Billings, 1999). Of all agents to be created for the game, *Iocaine Powder* has proven to be a very successful competitor, written by Dan Egnor. The implementation is relatively simple to understand, but is far from trivial. Six strategies are employed, all using direct prediction; from assuming an opponent predicts a prior prediction and acting accordingly (double guessing), to predicting subsequent triple guessing on the part of the opponent. These strategies are selected through a combination of frequency analysis and pattern matching (Egnor, 1999). The player also formulates that if an opponent is defeating the player

with any regularity, then the player will revert to a random strategy to ruin any opponent's modeling approach.

2.1.5 Pathfinding

The A-Star (A*) search algorithm (a development of Dijkstra's algorithm developed for graph traversal) is widely used in game environments for path finding, being particularly more reliable than a Neuroevolutionary approach (Hart et al., 1968). The performance of A* is improved over the Dijkstra algorithm due to its use of heuristics. The search uses a heuristic based upon two factors: the cost of a move between nodes g(x) (which takes into account the total cost from the start node), and a distance heuristic f(x) (from the node to the goal). The algorithm performs a 'best-first' search along a list of prioritised 'open' nodes with respect to the given heuristic. This guarantees a shortest path from the starting node to the goal, provided that the heuristic estimates never overestimate the distance to the goal (Russell & Norvig, 1995). If the position of the goal is unknown, any distance heuristic would be useless; in this case the Dijkstra algorithm would be the preferred method (as Dijkstra is essentially A* with heuristics returning 0).

Research has attempted to model the path making behaviours of humans in real-world situations (Krumm, 2006). The investigation into the predictive nature of human behaviour has shown startling results as to the predictability of human pattern analysis, identifying that 93% of human behaviour in relation to navigation is predictable (Song et al., 2010). The application of such a predictive technique across

virtual environments is many-fold. In this vein, military prediction of enemy units has been deeply researched, particularly into the potential destinations of naval units (Zhao et al., 2004) (Brown & Gordon, 2005).

In the context of games, path modeling is primarily concerned with online environments, where Dead Reckoning (a technique which utilises the current course vector and velocity to determine the next possible position) is used to model prospective agent position in an attempt to reduce network traffic caused by the transmission of positional data (Li et al., 2008) (Hladky & Bulitko, 2008).

2.2 Opponent Modeling

Modeling the intentions of other people is arguably one of the most important, yet most complex, of human abilities. An 'average' human is able to distinguish, through observation, the nature and intention of actions, as well as assess the capabilities of those who perform the action. As Sun Tzu (6th century BC) stated:

"If you know your enemies and know yourself, you will not be imperiled in a hundred battles... if you do not know your enemies nor yourself, you will be imperiled in every single battle."

2.2.1 Human Behaviour

Humans have an inherent ability to create a mental model of interactive behaviours, where the model is an approximated simulation of how another will behave (Johnson-Laird, 1983). These models are generally created through social observation and

interaction which yields much information from which to learn. Through the generalization and interpretation of the modelled set of actions over a variety circumstances, humans are able to display a socially acceptable interactive behaviour pattern (Byrnes, 2001); indeed it is arguable that the ability to construct models of others actions and intentions is central to the definition of what is “acceptable” in social behaviour. Le Doux (Le Doux, 1997) explains that through evolution, human instincts have been somewhat ‘hard-wired’, culminating in a set of adaptive behaviours crucial to survival. Le Doux argues that humans possess a fast mental process that warns of a potential danger, in addition to a slower reasoning process that models the nature of stimuli which initially triggered the fast process. For researchers and game developers alike, the understanding of interactions, and subsequent reactions (in a human manner), is essential to achieving intelligent, interactive behaviour. The ability of a machine to display human-like intelligence has been previously defined through the Turing test (Turing, 1950), where a machine attempts to convince a judge that it is actually human through conversation. The Turing test has been denounced by Livingston, claiming that the vague nature of the test is not representative of the ability of a human to be fooled by a machine, partly because the Judge actually *knows* that they are taking part in a Turing test (Livingston, 2006).

2.2.2 Opponent Modeling in Game Playing

Laird created 'Quakebot' as an opponent modeling agent for the game of QUAKE II (id Software 1997). 'Anticipation' is implemented into the agent to predict the actions that an opponent could take. It is noted that formulating a plan in order to defeat the opponent or use their situation to an advantage is a very arduous task, especially in the face of the numerous events that would have to be taken into consideration. Search techniques cannot be used due to branching factor, especially in terms of the numerous actions available. An inherent problem of anticipative modeling is that predicting an opponent's moves can take some time, which in a fast-paced game such as QUAKE II is a significant disadvantage. The agent is susceptible to being attacked, and the length of time in predicting impedes upon the time available to perform a suitable action. The authors answer this with a technique called 'chunking' which creates and saves a specific rule that the agent has inferred, to save time in regenerating the prediction at a later date (Laird, 2001).

Steffens covers opponent modeling within a multi-agent environment - the RoboCup tournament (Steffens, 2003). His work covers case-based reasoning (CBR), and how its predicative accuracy is dependent upon similarity measures between current and pre-programmed cases, in order to model the opponent. The RoboCup tournament consists of autonomous agents connected to a server playing a game of football (or soccer, dependent upon the territory). After receiving velocity information about all items on the field, such as players and the ball, the agents can execute discrete actions (Dash, Turn, and Kick), which are combined to create higher-level actions to represent

in game strategies, or an overall tactic. From defined situations, the predictions on an opponent's next action use game-state comparisons to a higher-level goal. An example is '*Shoot on Goal*' where if it is predicted that an opposing player with the ball will shoot, countermeasures are taken to prevent it occurring. Match history has also proven to be a useful tool in the evaluation and evolution of RoboCup teams (Nakashima et al., 2006). Their approach evolved strings of integers which represented a particular play strategy. The evaluation used match statistics such as goals for and against the team, and successful and failed incidents during the match.

Van den Herik and Donkers investigate the use of opponent modeling in commercial video games, especially in relation to improving the game playing experience (van den Herik & Donkers, 2005). The use of an opponent model to adapt the difficulty of games has been investigated by Spronck and van den Herik through a tutoring system (Spronck & Herik, 2005). Tutoring systems are generally used to introduce a player to a game, and help the computer gauge the strength of the player. The authors describe enhancements that could be applied in commercial games, attempting to create an even balance of gameplay so that experienced players should be challenged, and novice players should be catered for, not necessarily by dumbing down opponents, but making them use weaker strategies so that the opponent is not 'playing dumb', or making a mistake to pander to the novice player's weak play. Ideally a challenging agent would lose as many instances of confrontation as it would likewise win. The authors point out that there are some issues with current games, such that a difficulty

selection is too coarse due to its limited selection, as well as having been selected by the player themselves even though the player cannot yet grade their playing ability. The tutoring system gradually alters the strategies available to agents, whereas game difficulties are generally limited in scope, affecting enemies' strength rather than changing their tactics.

Spronck et al's Dynamic Scripting uses a reinforcement learning approach, allowing an agent to learn a game policy through interaction with its environment, and can then learn what to do to achieve its specific goal (Spronck et al., 2006). This is achieved by giving the agent a reward signal in relation to its success in the environment. Initially Dynamic Scripting was slow to react to opponents in some situations (although effective), and attempts were made to improve the performance of the scripting technique (Spronck et al., 2004). The improvements used penalty balancing by relating the size of a penalty to the current reward, keeping the sum of weights for each tactic constant, and being able to roll back tactic scores. Further Investigations use the Evolutionary State-based Tactics Generator (ESTG), which utilises an evolutionary algorithm to select the tactics for scripting automatically (Ponsen et al., 2006). The Dynamic Scripting approach is applied to Wargus, a Real-Time Strategy (RTS) game that is in a similar vein to Warcraft II (Blizzard Entertainment 1995). It is noted that for the previous version of Dynamic Scripting that separate tactics needed to be created by hand for Dynamic Scripting to choose from, and therefore find the most suitable approach against the current opponent. ESTG, however, creates these tactics

automatically. AI in RTS games is usually encoded in scripts, which determine all AI decisions over the course of the game. The flaws that can exist within Dynamic Scripting are such that although the agent adapts relatively quickly, all of the tests are done against an opponent multiple times to learn their strategy. This may not always be possible within a given game environment, where there may not be enough iterations of a situation that can give an accurate enough description of how to defeat the opponent.

2.3 Poker

Texas Hold'em Poker is arguably the most popular form of Poker played around the world today overtaking other forms of Poker as the most popular in casinos worldwide (Clark, 2006). The popularity boost of Texas Hold'em is such that the game has permeated into other forms of popular culture; In the Ian Fleming novel 'Casino Royale' for example (Fleming, 1953), British agent James Bond plays *Baccarat* against the primary antagonist, Le Chiffre. In the most recent adaptation of the novel into film, however, the game has been changed to Texas Hold'em, primarily because of the worldwide popularity the game has found (MGM, 2006). The three main actions performed in-game are common to all forms of Poker, as follows:

- ***Bet/Raise***: Add money to the pot, and increase the monetary risk for the bettor and the opponents.

- ***Check/Call***: Make the smallest bet required to stay in the hand (which may be nothing).
- ***Fold***: Take no further part in the proceedings of the hand.

These basic actions are an essential staple of all Poker games. It is the underlying strategy behind the decision-making process of a player that makes the game of Poker arguably one of the most skilful card games in the world. The complexity of Poker results largely from the fact that the only information available to a player of the game's state is that of their card(s) held, the community cards, and that of any past actions the opponents have made. Arguably the most important publication with respect to Poker-playing strategy is Sklansky's "The Theory of Poker" (Sklansky, 1992), which is widely considered by professional Poker players to be the best source of information on how best to play Poker. Sklansky discusses game playing concepts over various forms of the game of Poker, from topics such as bluffing (and other deceptive plays) to the psychology of the game and the importance of playing position.

As well as being a popular game, Poker has become an excellent testbed for research into games of imperfect information. Poker had mostly been overshadowed in the past in favour of games such as Chess. This is arguably due to the difficulty (and combinatorial explosion) that results from having to deal with imperfect information.

2.3.1 Early Work

Poker has driven numerous research efforts for many years, early efforts including Findler's research into machine cognition using Poker; judging that dynamic or

opponent-adaptive play is necessary in order to be successful, and that static play styles can be easily beaten once the opponent's style has been learnt (Findler, 1977). Earlier investigations into card game strategy have also considered the game of *Goofspiel*, which is deemed to have great tactical depth (Ross, 1971). Several approaches to understanding the mechanics behind games of imperfect information, however, have been based upon simplified variants of Poker: von Neumann used a simple form of Poker in his treatment of Game Theory (von Neumann & Morgenstern, 1944) as did John Nash and Lloyd Shapley (Nash & Shapley, 1950). Kuhn's approach to a game-theoretic analysis of Poker is also upon a simple game for two players, consisting of a three card deck, one card hands, and a maximum of two bets per player (Kuhn, 1950). Furthermore, Sakaguchi used three simple forms of Poker to investigate recommended changes in action selection based upon the number of opponents (Sakaguchi & Sakai, 1992). More recently, many researchers have shown interest into Poker playing in general, most notably the GAMES (Game-playing, Analytical methods, Minimax search and Empirical Studies) research group at the University of Alberta which has released numerous papers on Poker research. Darse Billings, once a professional Poker player, has aided and assisted in the creation of Loki, a Poker-playing AI (Billings et al., 1998). The variation of Poker in question is Texas Hold'em Poker. Billings et al describe the system and how it models opponents, and then evaluates its performance against other artificial Poker agents. Their research spotted that in previous iterations tested against humans, the agent performed well initially, until the human players changed their strategy, to which the agent could not adapt.

The authors list some ‘requirements’ that a world-class Poker player would have to fulfil in order to be successful in a game of Poker (Billings et al., 1998). It is noted that all these requirements are intertwined, and should not be held separate from one another. The requirements state that a player must be able to have a good gauge of hand strength, whether it is as simple as an assessment of your own cards and community ones, or a more complex assessment of your cards, table position, and prospective opponent hands. Having a good sense of how the hand obtained can either improve or depreciate dependent upon the upcoming turn cards is also a definite requirement. The betting strategy encompasses all of the requirements, requiring a gauge of hand strength, potential, and winning chances. It is noted that unpredictability and bluffing are two very important factors for a Poker player, of which it is important that the potential of a hand improving be taken into consideration; blindly bluffing with a poor hand is much worse than bluffing with one that could turn profitable on the next turn card. Finally, as part of these requirements, it is noted that opponent modeling is extremely important, as hidden information, such as an opponent’s hidden cards can more easily be predicted when analysing and interpreting as much data that can be garnered from an opponent’s actions, which include betting plays, and betting amounts.

2.3.2 Game Theoretic Approaches

It has been noted that an optimal strategy in terms of a game such as rock-paper-scissors (RoShamBo) is a random one, and Daphne Koller and Avi Pfeffer note that for a

game such as Poker there must be an optimal random strategy, just as there exists an optimal deterministic one for a game such as Chess (Koller & Pfeffer, 1995). At the time AI research had all but ignored imperfect information games (Nicholas Findler's work notwithstanding), due to the complication of modeling prospective circumstances. The authors introduce the *Gala* system which provides a language, much in the vein of Prolog, to describe game situations, and derive an optimal strategy through a self-generated game tree. It is noted that the 'paradigm' for solving games involved transferring the game tree to a matrix, as a *standard* or *strategic* form. This is exponential in size in relation to the size of the game tree. Koller and Pfeffer (1995) use a simplified form of Poker in order to test the Gala system; a deck of three cards is used, and a single card is dealt to each of two players, betting rounds are initialised where when all bets are equal, or one player has decided not to bet, the betting player or the player with the highest card are the winner. The authors note that any deterministic strategy in this situation would not be prudent, as deterministic play spawns predictable play, which in turn can be easily defeated once predicted. The Gala system is tested against other systems which attempt to perform the same strategy optimisations, and while Gala runs in linear time, other approaches take much longer, even though Gala generates an equally optimal strategy to the exponential-time attempts. The results also give very interesting strategies in relation to betting probability for the dealer and gambler in an 8-card game of Poker. The authors note that a game such as full-scale Poker, with the extremely large size of the game tree would prove impossible to generate a solution for, and that it is unlikely that a solution

is ever to be found (Koller & Pfeffer, 1997). Billings et al comment on the work by Koller and Pfeffer, stating that although a game-theoretic optimum strategy could be employed, if quantifying such a game tree was more easily accomplished, it would not help towards 'solving' the game, due to the necessity of modeling an opponent's behaviours. This necessity is defined by the fact that optimal behaviour within an imperfect information game is useful, but within a game such as Poker, where bluffing is prevalent, an optimal strategy will not maximise winnings (Billings et al., 1998). An opponent's bet could represent a bluffing move or a show of confidence, this being difficult for any player to determine. Part (if not most) of the challenge of Poker is the understanding of your opponent, recognizing and exploiting weaknesses and subtle changes in their behaviour. A good human player has weapons in his arsenal unavailable to the current generation of AI players, using external factors such as conversation and facial/bodily gestures, which can be used to transmit (mis)information to the opponents. This author feels that a single 'solution' for Poker is unattainable, primarily due to the stochastic nature of the game as well as factors such as the aforementioned *human* aspects of the game. Contrary to the fact that although Checkers is solved, the game is still interesting for human players to play recreationally, whereas a solution to the game of Poker (where an optimal strategy could be more easily learnt) would effectively destroy the competitive nature of the game. Billings et al's described Loki model uses an unsophisticated betting strategy, where betting actions are performed relative to hand strength and win probability; this is also utilised to determine the amount of money that should be bet (a function of bet

probability and current pot amount). The opponent is modeled directly using betting actions, and each action updates the weights of each hand combination (effectively the probability that an opponent holds such a hand). Each opponent's action is classified by the action taken, the amount of money bet if any, and the stage of the hand in which the action took place. For each opponent action made, the weights are transformed to represent the analysis changes. Loki is tested against multiple simple artificial agents, some which used a modeling approach, and some which did not. The results clearly show that a modeling approach quickly beats an approach which does not. Billings et al 1998 claims that further research is to be aimed at creating an adaptive player, explaining that the current build will not adapt to a player that changes his style over time.

(Billings et al., 1999) adds more sophisticated changes to the process of determining an opponent's action as well as suggesting a suitable action to take. This is performed using probability triples, where each probability (f, c, r) represents the probability of folding, checking, or raising, and can be applied to the suggestions for either the player or the opponent to infer probable, or highly likely successful play through updating weight tables. The improved system uses a simulation of possible plays, and calculates an expected return of the opponent taking a certain action. The simulation uses a selection of the highest ranked probable card holdings that each opponent has in order to have a suitable approximation of return. The paper shows a respectable improvement in the player's performance in comparison to previous implementations.

(Billings et al., 2003) covers the generation of game-theoretic optimal strategies for Poker, something which earlier papers commented was not necessary for playing Poker due to the nature of bluffing. The paper states that simpler variations of Poker do not fully quantify the complex nature of full-scale Texas Hold'em, and proposes using *pseudo-optimal* solutions by reducing the game in a less restrictive manner (such as Burns (2006) which investigates the optimality of commonsense poker strategies, uses a deck in which each player is dealt a card classed as 'high' or 'low'), although the authors admit that there is no guarantee that these reductions will lead to reasonable predictions. These suggestions included reducing the number of betting rounds, in order to reduce the length of the game tree. The resulting player, designed to use only the game tree to play, with no opponent modeling methods, was able to perform adequately against a world-championship level player, although after 4000 hands the professional understood the style of the player, and the agent was unable to come back from persistent losses. It was noted that the addition of opponent modeling would provide an extremely strong Poker player. The performance was not necessarily the important point, however, as the paper was the first successful attempt at approximating a game theoretic strategy for a full-scale Poker game. Further game-theoretic approaches in the realm of Poker have considered the idea of regret minimization, where as the regret of a strategy in response to an opponent strategy approaches zero, the strategies approach a Nash equilibrium (Zinkevich et al., 2008). In the pursuit of regret minimization, various techniques have been explored such as

counter-strategy evaluation (Johanson & Bowling, 2009), imperfect recall (Waugh et al., 2009), and Monte Carlo Sampling (Lanctot et al., 2009) (Ponsen et al., 2010).

2.3.3 Opponent Modeling and Poker

Aaron Davidson, along with Darse Billings, Jonathan Schaeffer and Duane Szafron adapted the Loki AI (now called Poki) with the interest of creating further adaptive play with the use of Artificial Neural Networks as part of the modeling system (Billings et al., 2000). The argument for this is that use of an ANN will maximise the accuracy of the targeted output. The most recent previous action, and the previous amount to call were seen as the determining factors, and most important inputs of all tested. The training data for the network was based on betting actions of players on an Internet Relay Chat (IRC) Poker room. Using the ANN, the agent was subsequently able to predict future actions with 80% accuracy, compared to the 57% of the previous system.

In relation to creating an *adaptive* Poker player, Luigi Barone and Lyndon While started investigating approaches to evolving players to learn games of imperfect information, settling on Poker as a suitable testing ground (Barone & While, 1998). His research proceeded to investigate the evolution of a Poker player for a simplified variant of Poker (Barone & While, 1999). Barone and While point out what many believe to be four distinct styles of Poker player: *Loose Aggressive*, *Loose Passive*, *Tight Aggressive*, and *Tight Passive*. The authors also suggest similar player requirements to the previous Billings et al papers, namely hand strength analysis, position in play order,

and risk management. The authors have also used a bet tuple which represents the probabilities of betting folding and checking, also similar to the Billings et al paper. The authors use evolutionary algorithms to evolve a player from a randomly generated one to defeat the four playing styles. The paper continues to demonstrate that an evolutionary approach could be used, with some resolution, to adapt a player dynamically as opponent styles change. This research is further elaborated into the game of Texas Hold'em and shows how the player evolves due to differing styles of the opponent and adapts its strategy (Barone & While, 2000). This adaptation is quite slow, but shows much promise in the possible execution of a dynamically adapting player. Pieter Spronck's previously mentioned work (Spronck et al., 2004) also looked into constantly dynamically adjusting behaviour for a game-playing agent, but it avoided convergence through weight control, Barone's work avoids convergence by being negligent of domain-specific factors, such as play position. Barone shows that utilizing differing styles in the evolution of a player is beneficial to the performance of the final solution, and Kendall and Willdig (Kendall & Willdig, 2001) perform a similar approach to the evolution of a weighting factor to alter the decision process of a player against the same varying styles (which are discussed in further depth in Chapter 4) in the realm of draw Poker, where the players are dealt five cards before a round of betting, and can subsequently trade two cards before a final round of betting. Both Barone and While's (as well as Kendall and Willdig's) approaches use a single agent, which gradually adjusts its performance given the style of opponents at the table. Carter and Levine have also investigated the evolution of Poker players for tournament

Poker, noting that the performance of evolved agents is limited in its capacity due to the lack of an opponent modeling capability (Carter & Levine, 2007).

Texas Hold'em has proven to be probably the most investigated form of the game of Poker, and few have varied from such a path, but Saund developed an approach to analysing and utilising an opponents' betting behaviour in seven card stud Poker (Saund, 2006). The approach not only uses betting actions, and cards held in hand, but also allows for some usually hidden information, such as some downcards. This knowledge is then used to create a player that can further infer about a player's unknown downcards, much in the same way as Billings' earlier approach.

2.4 Summary

In this chapter, we have explored literature within the realms of game-playing AI starting from the initial game-theoretic approaches to games such as Chess, expand from this into the adaptations of the basic minimax search onto the games of Othello and Checkers. Research related to both Perfect and Imperfect information is considered and contrasted, especially with the increased difficulty in respect to stochastic environments compared to the deterministic nature of most board games. The importance of opponent modeling is investigated with respect to understanding human behaviour and how humans constantly model their environments.

We note the importance of opponent modeling with respect to both understanding an adversary for agent improvement, as well as that of sculpting an agent's play to

improve the game experience for a human. Finally, we discuss Poker, arguably an important testbed for opponent modeling research. We discuss some of the earlier approaches to how Poker was treated, as well as how the game is frequently simplified for use in research. These simplifications are almost completely necessary for any investigations into a game-theoretic approach into the game of Poker. Finally we discuss work where opponent models are applied in the realm of Poker, and how various techniques (such as evolutionary approaches) can be used in conjunction with opponent models to aid in creating adaptive Poker players.

Chapter 3: Agent Path Prediction

Predicting the path of an agent is a necessary technique in many facets of interaction. A simple social example is the navigation of a person through a crowd of people; the predictive capability of a human is required to determine the correct time at which to cross the path of another person. This chapter serves as an introduction to the applicability of Bayes' Theorem to the prediction of an opponent, as well as demonstrate the potential strength of Bayes' rule as an approach to aiding decision making (before moving onto the imperfect information environment of Poker in Chapter 4). We explain our approach to demonstrate a probabilistic technique for determining an agent's short-term goal in real-time evaluation for use in determining an appropriate and intelligent interceptive reaction. We use the A-Star (A*) algorithm as a terrain-analysis technique in order to determine action-goal probabilities, and apply Bayesian analysis upon the behaviour of an opponent with these probabilities in order to determine the potential goal of the opponent. The use of action probabilities in this respect makes our approach applicable to a wide range of varying situations, such as in a competitive environment where network (and positional) data is available for a number of human players.

3.1 Bayes' Rule

Bayes' rule relates conditional and marginal probability distributions of random variables, and shows that however different the probability of event A conditional

upon event B is to that of B conditional upon A, there is still a relationship between the two that allows one to be calculated from the other.

$$\Pr(A \mid B) = \frac{\Pr(B \mid A)\Pr(A)}{\Pr(B)} \quad (1)$$

Equation (1) gives Bayes' rule where, in a game scenario, our opponent has an unknown strategy, *A*. We need to determine the strategy *A*, given the observed set of opponent actions, *B*. The analysis of human action and inference with relation to a goal or strategy can be applied over numerous fields. For example, the imperfect information conveyed by players in Poker can be similar to that of partial information in a 'deathmatch' based game, where players are constantly unsure of the enemy position.

3.2 Agent Path Prediction

The prediction of the goals of an agent is an important tool in achieving believable reactive behaviour, and games give a greater opportunity to test and apply these approaches. In many games, forecasting an opponent's path is critical. A useful analogy can be drawn to deathmatch-based games, such as QUAKE III: ARENA (id Software 1999), where predicting the intended target of enemy movement gives an indication of where to fire. A 'deathmatch' is a scenario where two or more agents use various weapons (from melee weapons to pistols, rocket launchers, or fictional devices) in order to accumulate kills (or 'frags') within a time limit or a set limit of kills in order to win the

match. In this scenario, the 'splash damage' of certain explosive weapons can be employed as both a successful play tactic and also an indication (to the player) as to an agent's predictive capability, exemplifying the appearance of intelligence to a human player. Investigation into developing intelligent, human-like agents has been recently stimulated by the 2K Games-sponsored BotPrize Competition, which provides a Turing-test type environment where agents, a human confederate and a judge all inhabit the game world in a series of competitions using the UNREAL TOURNAMENT 2004 game (Hingston, 2009) (Epic Games 2004). This game is a deathmatch game similar to the aforementioned QUAKE III: ARENA. In order to promote competitive play by the human confederates, a minor prize is awarded to the best human player such that no 'abnormal' behaviour is exhibited by the human to try and convince judges that they are actually the 'bot'. So far, no competitor has taken the $7,000 prize offered to researchers who can develop an agent which convinces a judge that it is a human player (instead of the human confederate). One point of note is that this author played in the competition as a human confederate in 2008, winning the 'best human' award, convincing 4 out of the 5 judges that he was human.

3.2.1 GridWorld

We introduce GridWorld, a teaching/research tool for AI techniques, and harness for a competitive AI environment created by the *University of Bradford* and *Black Marble Ltd.* with sponsorship from *Microsoft* (Cowling, 2006). GridWorld is a platform for teaching AI programming and conducting research using the development and study of

AI in games, which draws analogies to real game environments through the use of a visual grid rather than the underlying grid that is utilized in commercial games. An example of *Halmoids*, a GridWorld game, can be seen in Figure 3.1.

3.2.2 Halmoids

The aim of Halmoids is to control one or more pirate ships in order to navigate to a treasure chest with the same colour corresponding to that of the ship(s), and the scoring is determined by the number of ships that have reached their goal. The in-game icons (designed by the author) are the following:

The Ship represents the unit that is controlled by the player or an agent

The Treasure represents the goal which the player needs to reach; each goal is colour-coded to each player's ships

The Lighthouse represents an impassable 'rock' object, used for determining the layout of the terrain

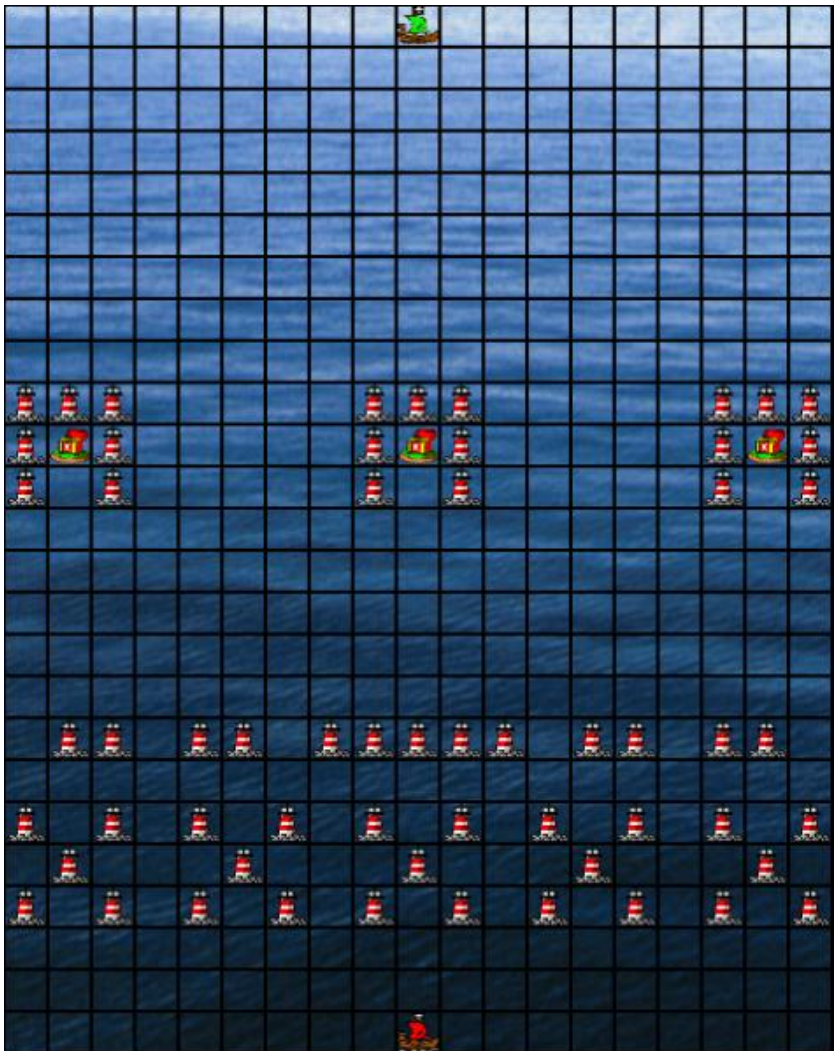


Figure 3.1: A simple ship-blocking terrain

In order to win, a player must maintain an advantage over their opponent, and therefore should attempt to stop one or more of their opponents' ships from reaching their goal. This game is somewhat analogous to a 'Capture-the-Flag' mode of play from many multiplayer First Person Shooter (FPS) games such as UNREAL TOURNAMENT 2004 (Epic Games 2004), where attacking or defensive strategies must be chosen in response to the opponent's actions. The interception of an opponent requires early inference of their destination, given that the opponent has multiple goals to choose

from. In this example, the lower ship has been given only one of the treasure chests as its target destination, and the upper ship needs to intercept the lower ship before it reaches its goal. A useful analogy can be drawn through the consideration of sports simulations, an example such as player 'marking' tactics within a soccer video game such as PRO EVOLUTION SOCCER which simulates the game of Football (Soccer) in International games and domestic leagues (Konami Corporation 2001). A 'marked' player must attempt to evade the opposition player assigned to follow and restrict his movement, especially in such scenarios as those of a free-kick or corner kick (Denzinger et al., 2005). Halmoids as a game represents the task of both the marked player, and the marker themselves. In this work we concentrate on the role of the marker. As with many game scenarios, the importance of predicting a player's tactical intention through action analysis is integral to successful play; to fail to understand the opponents target can potentially mean the loss of a goal in soccer, or the entire game in the case of Halmoids.

3.3 Terrain Analysis

An agent in a real-time game should be able to generalise its ability to infer destination over many scenarios or, in the case of Halmoids, terrains. We attempt to create a generalised opponent modeling approach through partial terrain analysis. The application of Bayes' rule requires an initial probability distribution to represent the link between opponent action and strategy/goal. This probability distribution can either be constructed from data of past interactions, or a designer's *subjective belief* of

how an opponent should behave to achieve its goal.

3.3.1 A* Analysis

In a situation where the nature of a terrain/opponent is unavailable, we cannot take a subjective viewpoint. Thus, we need to be able to generalise for unforeseen terrains, and actively procure environmental data before analysis can be performed. In this chapter, we use A* search with a Euclidean distance heuristic to create paths from our opponent's controlled unit to each of its potential destinations, as the Euclidean heuristic never overestimates. To explicitly compare our opponent's position each turn in comparison to the paths we create would be a poor decision, in case our (human or otherwise) opponent uses a different distance heuristic or another (presumably non-optimal) form of pathfinding. This is similarly the case if a single path is required by more than one goal, which can potentially apply in the case of 3.1. In light of this, our approach is to analyse each position along the path, in relation to the previous position. Since our environment is grid-based, we can count for each move along the path, a record of the direction moved in. From this, we can calculate a probabilistic relation of which direction would be moved in given a potential destination. The set *S* of directional probabilities per path is recorded, and is represented by

$$S = \{P(U), P(D), P(L), .., P(DL), P(DR)\}$$

such that

$$\sum_{n=0}^{n} P(n) = 1 \qquad (2)$$

Where U represents the Up direction, D represents the Down direction; L represents the Left direction, and R represents the Right direction. For each move along the path in each direction, a per-direction counter is incremented by 1 for that given direction. Obviously, this can be transformed to any directional or angular scale of choice, but these eight directions are chosen to make the technique applicable with the grid-based environment. This directional choice could potentially also apply to the spatial representation of a character in both 2 and 3-dimensional environments.

3.3.2 Potential for Probabilistic Error

A limiting factor upon the accuracy of our observation is that of unexpected behaviour; if an opponent makes a move that is not anticipated through our 'absolute' prediction of the path to the potential target, the narrow probability distributions can potentially mean an erroneous analysis. Many values could be given a very low probability value since the A* analysis of the terrain may have yielded no instances of this action.

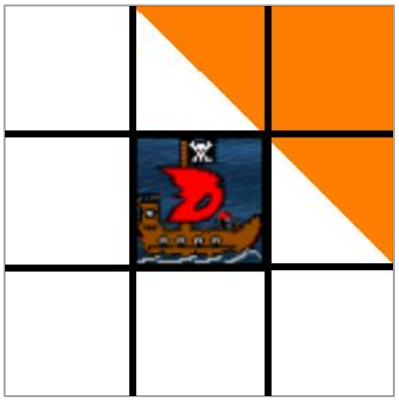


Figure 3.2: Representation of Move counter in relation to related motion direction

To allow for the consideration of deceptive or non-optimal pathfinding behaviour by the opponent, we count a move in each of the eight directions as a single step in the original direction, but also add a half step to the count for each neighbouring direction. E.g. if the path we assume for the opponent takes a step in the UP direction, our count of the moves in the direction are incremented by 1, but we also increment UL and UR by 0.5. This provides a 'buffer' for potentially unexpected (or non-optimal) moves. We now have a set of per-directional probability distributions for each possible destination. Using the terrain from

Figure 3.1, we generate a distribution as shown in Table 3.1. The probability distribution would be much more widespread across all directional values if a more complex terrain was to be considered, whereas in this case we can see that values are fairly similar across many targets due to the symmetry of the terrain.

	Target 1	Target 2	Target 3
UP	0.3653	0.3845	0.3653
DOWN	0.0003	0.0003	0.0003
LEFT	0.1346	0.0577	0.0003
RIGHT	0.0003	0.0577	0.1346
UP-LEFT	0.3845	0.2499	0.1153
UP-RIGHT	0.1153	0.2499	0.3845
DOWN-LEFT	0.0003	0.0003	0.0003
DOWN-RIGHT	0.0003	0.0003	0.0003

Table 3.1: Probability distribution over all possible actions per available opponent targets from Figure 3.2.

Given our initial probability distribution, we can now perform an iterative calculation using Equation (1), which will take each action observed by the player and gain a posterior probability distribution of our opponent's most probable target, which our opponent chooses at random. As our observation yields further information as to our opponent's target, we can then intercept using the appropriate path to block our opponent's progression. Figure 3.3 shows the convergence of probabilities given the movement data of an opponent moving to the right-hand treasure chest, *Target 3*. This shows that the use of a probability distribution over the predicted A* path can prove successful. Figure 3.4, however, shows the performance of a player moving to the central treasure chest, Target 2. As is shown through observation of the terrain defined by Figure 3.1, to move to the central chest requires the movement along one of the paths for either adjacent chest, hence the initial convergence of target belief being upon that of Target 3. The change denoted by the increase of the belief representing

that of Target 2 can be explained through Figure 3.5, which shows a map of the reaction of the Bayes-controlled ship (*b*) to the opponent action set. As can be observed, *b* successfully intercepts the opponent before it reaches the target. Observing the path taken by *b*, we can see that due to the frequency of moves to the right-hand treasure chest, the path before move number 9 (where our analysis determines a change in target) is at a point where the belief that the opponent will move to the right-hand treasure chest is strong enough to cause an effect that displays the realisation upon the next move that its initial belief is incorrect. This performance yields a behaviour that appears human in its folly, as well as showing enough intelligence in order to correct its error.

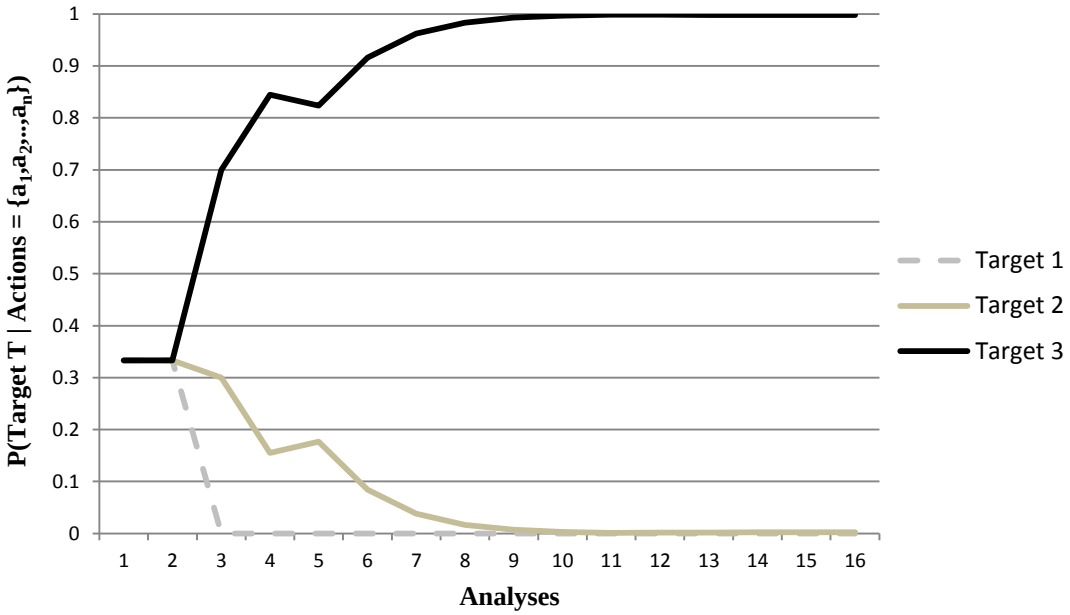


Figure 3.3: Bayesian analysis of an opponent ship heading towards Target 3 (right-hand treasure chest)

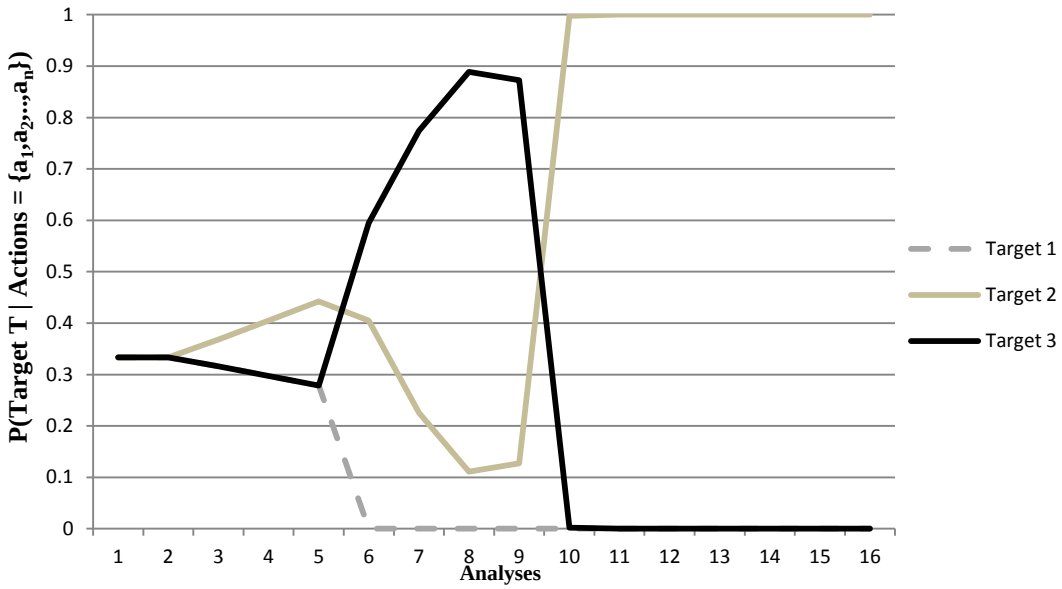


Figure 3.4: Bayesian analysis of an opponent ship heading towards Target 2 (central treasure chest)

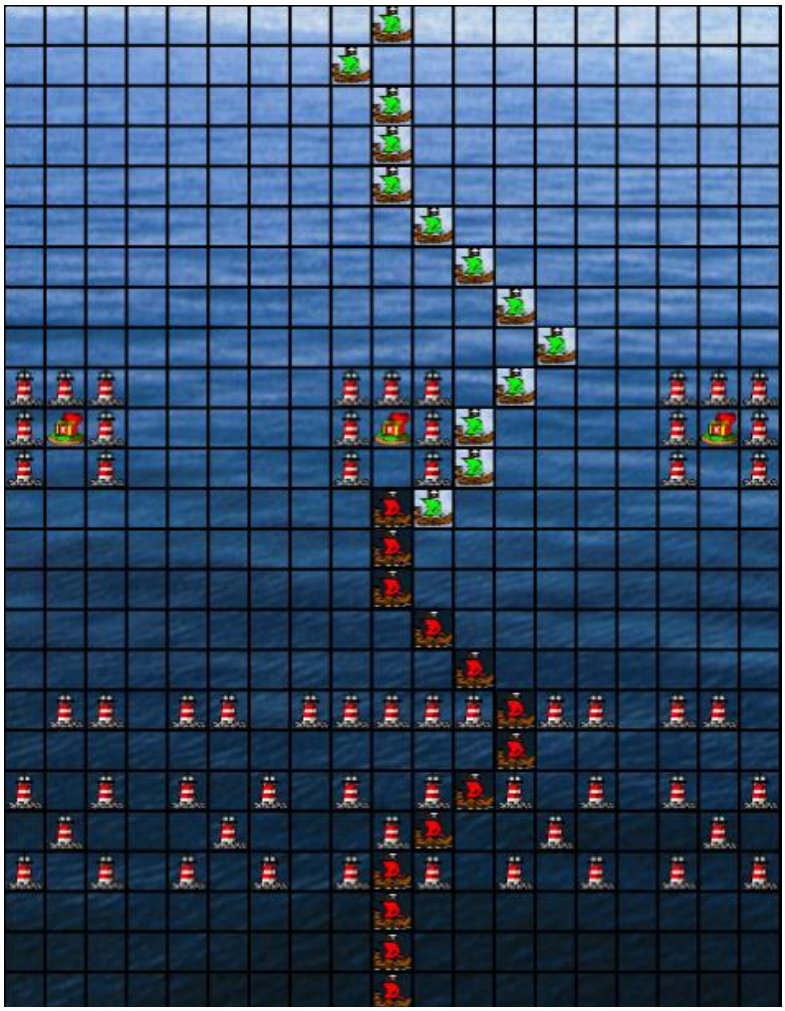


Figure 3.5: Performance of the Bayes' Rule -Controlled (Green/upper) ship

3.4 Coping with Random Behaviour

In order to test the robustness of the A*-Bayes' Rule hybrid, varying levels of random behaviour are added to the lower ship's pathfinding, so that with probability *p* the move made is at random rather than following its original path. Figure 3.6 shows the accuracy of our analysis against an opponent where $0.01 \leq p \leq 1.0$, applied in increments of 0.01.

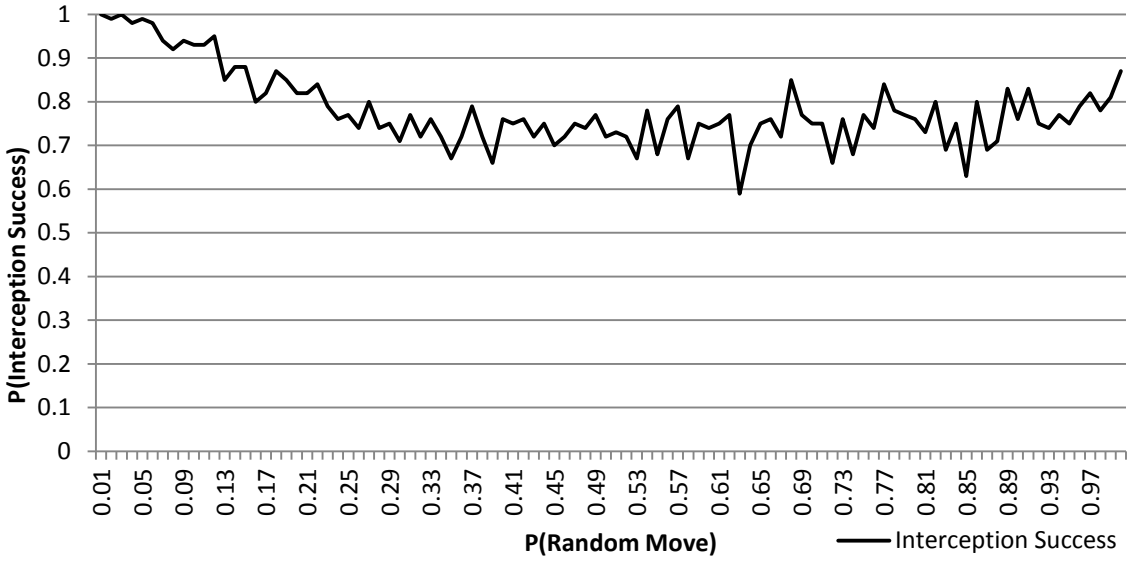


Figure 3.6: Interception success of a Bayes' Rule-controlled ship against an opponent applying varying levels of randomness to its actions

As we can observe, the predictive accuracy falls with the gradual increase in the amount of random behaviour displayed by the opponent. This can be accounted for by the presence of moves which are linked with a very small probability in relation to the target, as displayed in Table 3.1. This is further compounded by the amount of random movement that occasionally forces a 'bluff' move towards a different goal at the last minute; this sidestep can occasionally be enough to fool the analysis into believing in a

change in destination. We can also see that as the amount of random behaviour further increases, the success of the analysis increases somewhat; this is due to the lack of focus or direction for the randomised player, causing an erratic approach, stopping or slowing the player from reaching its goal.

3.5 Summary

This chapter investigates the use of opponent modeling in a real-time game with some of the characteristics of a competitive video game, and has described how a probabilistic A* path analysis can be used as a generalizable means to develop probability distributions representing the link between assumed opponent actions and a short-term goal. Consider a first person shooter (FPS) type game where the player has a choice between finding cover, moving towards a health pick-up, or escaping through a given exit; applying a collection of action probabilities to various directions of movement and other actions (in relation to potential goals) can yield a good idea of the player's tactics. Considering various other (video) game environments, tactical goals are prevalent in determining the success of an agent (or player) in many genres of game – Real-time strategies, that require the employ of planning and tactical decisions, may be of particular interest; given a selection of strategies, such as the human case base used by Louis and Miles in their customized strategy game (Miles & Louis, 2005) (Louis & Miles, 2005), we may analyse the use of independent actions (as a chain of actions or a set) to determine the overall opponent goal. A further example may be of a fighting game, such as KING OF FIGHTERS XII (SNK Playmore 2009) or STREET

FIGHTER IV (Capcom 2008), where sequences of moves (known as combinations, or 'combos') or special abilities (such as throwing a fireball, or any particularly damaging move) can be employed to drain the 'energy bar' of another opponent. In the situation where enemy inputs can be observed, such as the GoCap system (Alexander, 2002), the set of actions input to perform a move can be used to observe the intended attack, and resultantly perform a blocking or countering action in time. In these cases, the agent would be able to react believably (given a suitably coded response) to the player's decision. At a higher level, initially low level strategies could be related to much grander strategic ideas – we only need to know action probabilities for hand-coded strategies in relation to observed action probabilities. A player which is guessing opponent strategies (and changing guesses in response to opponent actions) could then appear to have a consistent, and human-like, approach to thwarting opponent strategies, even when these strategies will rarely be known in advance. We have shown in this chapter that Bayesian analysis upon this distribution in relation to opponent actions observed in real time can determine the goal of a previously unseen opponent, and hence determine the point of interception. The potential for applying Bayesian analysis and, by relation, opponent modeling to game agents is considerable; the data that can be gleaned from human interaction is expansive in scope, yet few have attempted to use any means to analyse it and apply any inference to a commercial game environment. It can be argued that Intelligence is only recognised and defined subjectively; the capability to behave in an intelligently believable way is enough to create an immersive, interactive experience. The weakness in this approach

is the time taken to converge to a conclusion of which goal an opponent is heading towards (and the potential for deceptive play to reduce the accuracy of the prediction), but a human's predictive capability where unpredictability and human error could similarly contribute to a delay in an accurate belief of opponent destination. Furthermore, knowledge of the strategy (or at very least the potential goal) is required for an accurate analysis to be made – if we consider that the opponent's target is that of a goal, but of something else, possibly a strategically advantageous position, then the analysis could be erroneous. However, in chapter 5 it is shown (using a simplified Poker game) that even if the actual strategy of the opponent is not strictly defined within the set of opponent strategies, an approximation of the potential strategy by a modeling technique is still inherently useful. This can be analogous to a human approach to game-playing, where approximations of an opponent (however incorrect) can still prove useful.

Chapter 4: Strategy Recognition in Simplified Poker

In this chapter we investigate the application of opponent modeling in a simplified form of Poker; a 10-card deck, a single-card per player, highest card wins scenario. We describe four types of opponent agent in relation to action frequencies and determine 'anti-' strategies that are capable of beating each type to an adequate degree. We subsequently use Bayesian analysis upon our opponents to determine the style of strategy used given only the actions performed by the opponent in game. This analysis then determines which anti-style should be adopted by the player in order to most successfully counter the opponent's play style.

4.1 One Card Poker

We use a simple version of Poker, which still maintains some of the tactical 'flavour' of a full-scale Poker game, but is more amenable to experimentation. The deck consists of ten cards, numbered 1, 2, …, 10 (names and suit are arbitrary, only the strength order of the cards is important). Each player has an initial credit of 10 chips, and each hand entered requires a one-chip ante from each player, after which each player is dealt one card. This approach is somewhat richer than that of Koller and Pfeffer (Koller & Pfeffer, 1997), which uses an 8-card deck to find an optimal mixed strategy using game theory with each player having only one card and one chip each, and Burns (Burns, 2006), which investigates the optimality of common sense Poker strategies,

using a deck in which each player is dealt a card classed as ‘high’ or ‘low’. In our game, the winner of the hand is the player with the highest valued card at the showdown at the end of each hand.

4.1.1 Game Rules

After the cards are dealt, the players take turns in a clockwise direction, and make a decision whether to *bet*, *fold*, or *check*, given the value of their card. Betting (which is equivalent to a ‘raise’ action), and each subsequent raise costs one chip. Once all players have matched one another’s bets (or all but one player has folded) the showdown is reached, and the player with the highest card (or only player remaining) receives the pot. The players continue playing further hands until there exists a tournament winner who has won all of the chips. In this current work, a bet limit of 4 chips per player per hand is employed. This limit prevents a tournament ending in a single hand, as could potentially happen in a ‘no-limit’ game as well as preventing a player from going ‘all-in’ (betting all held chips, and creating a side-pot in a multi-player environment) to avoid other players from being unable to match any large bet, as well as potentially ending a tournament in a single action. Primarily, however, these limitations have been applied to allow us to observe opponent actions rather than tournaments ending prematurely. The strategies which can be employed in this version of Poker, particularly of bluffing and opponent modeling, echo those of a single betting round of the full-scale game.

4.2 The AI Players

4.2.1 Distinct Style Players

Poker players may usefully be categorized into four main styles:

- **Loose Aggressive (LA):** A player that typically over-values hand strength, who will constantly force the pot higher, even with a relatively weak hand.
- **Loose Passive (LP):** A player that will also over value their hand, but will generally call, and only bet when they believe that they are likely to win the hand.
- **Tight Aggressive (TA):** A player that accurately values their card, and will fold more often, but any hand where a high card is held, then the player will bet aggressively.
- **Tight Passive (TP):** A player that plays very few hands, and even when doing so will generally call, and only bet in rare situations when a win is most likely.

Barone and While recognized these play styles as part of their investigation into evolutionary adaptive Poker play, and have also been utilized as part of Kendall and Willdig's work (Barone & While, 1998), (Barone & While, 1999), (Barone & While, 2000), (Kendall & Willdig, 2001). These classifications are used as standard Poker terms for describing in a general, simplified, sense how a player plays his hands, although a good human player would generally use more than one of these styles (possibly varying between them from hand to hand). Each of these styles of player was created using a simple deterministic design (Figure 4.1). Having four precisely defined player

styles gives us a good basis for analysis, even if the problems of analysis are easier than for the full game of Poker.

4.2.2 Implementing the Distinct Styles

A player's style is characterized by a probability pair (α, β), where α represents the minimum win probability (the probability this player has the best hand) required for a player to remain in the hand, and β represents the minimum win probability for the player to bet. Then α is responsible for whether a player is tight or loose, and β determines whether a player is passive or aggressive. If the win probability is less than α, the player will make a checking action if no money needs to be placed in the pot to remain in the hand, and fold otherwise. It should be noted that these players act on card strength alone, and ignore opponents' betting actions.

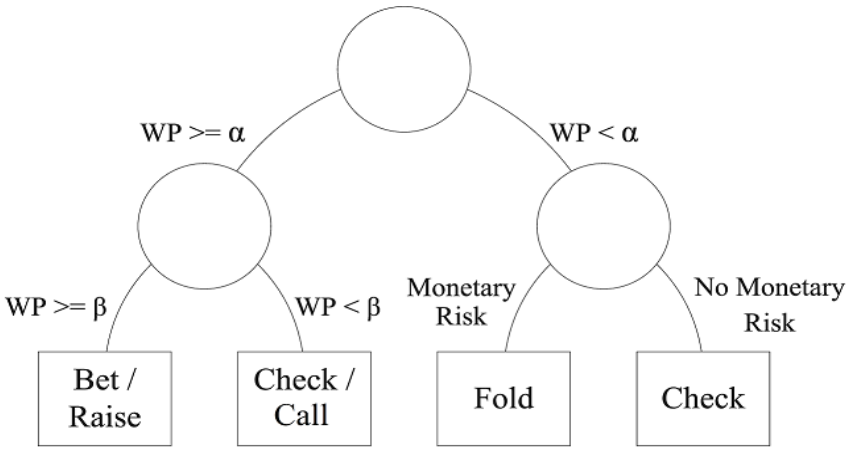


Figure 4.1: Layout of a General Player

A pair [α, β] represents a mixed strategy, with a distinct play style. The α and β values for each playing style are defined in Table 4.1. These values have been determined subjectively by the author based upon the descriptions laid out at the start of this

section. The play frequencies displayed in Table 4.3 infer that the α and β values here yield the desired playing behaviour for each of the individual styles.

	A	B
LA	0.1	0.2
LP	0.1	0.9
TA	0.5	0.6
TP	0.5	0.9

Table 4.1: α and β values for each style of deterministic player.

4.2.3 Design of the Anti-Players

"Anti-Players" were created as a 'nemesis' to each of the LA, LP, TA, and TP players, still based upon the same two-parameter model of playing style displayed in Figure 4.1. The values of α and β for an "Anti" Player are dependent upon the number of opponents, and the respective styles of those opponents. (α, β) pairs were tested in increments of 0.1 for $0 \leq \alpha \leq \beta \leq 1$ to determine the best values. Each (α, β) pair is tested in a 100-game heads-up tournament, with all players starting each tournament with 10 chips. For example, Figure 4.4 gives the performance of different [α, β] pairs against three LA players.

Table 4.2 gives the [α, β] values for the Anti-LA, Anti-LP, Anti-TA, and Anti-TP players, with the success rate of these values against the four distinct styles. Against loose players, the α and β values represent a rationally tight style of play, as loose players will often squander chips when facing a tight opponent that holds a strong card. When playing against tight players however, a loose strategy is adopted to remain in play, and a tight one in relation to betting/raising. This appears rational, as many tight

players will fold when holding a weak card, possibly leaving the pot to a looser player that may hold a weaker card. The tightness in relation to betting is also rational; staying in hands where the pot is small is wise, but when a strong card is held, the player should try to raise the pot as high as possible. When against loose players, the win percentage cannot reach 100% due to the situation where the loose player has the highest card and each player bets, ultimately costing the Anti-Player all of its chips. Conversely, when facing tight players, the win percentage cannot reach 100% due to the Anti-Player remaining in most hands, meaning that in some circumstances, the player will run out of chips due to the ante per hand. It should be noted, however, that the usage of tight and passive play is somewhat exaggerated in our two parameter model, when compared to that seen in real Poker (Sklansky, 1992).

	α	**β**	**WIN %**
Anti – LA	0.6	0.8	76
Anti – LP	0.8	0.9	63
Anti – TA	0.0	0.7	71
Anti – TP	0.0	0.8	75

Table 4.2: α and β values for each style of Anti-Player.

4.2.4 Modeling the Opponent using Bayes' Rule

Having a means of defeating each style of player raises the question as to whether we can intelligently select between them to create a player that could defeat all types and combinations of opponent. We propose that Bayes' theorem could be used to

analyse past play information (history of each player's betting actions), in order to determine the style of each opponent.

$$\Pr(A \mid B) = \frac{\Pr(B \mid A)\Pr(A)}{\Pr(B)} = \frac{\Pr(B \mid A)\Pr(A)}{\sum_{a} \Pr(B \mid a)\Pr(a)} \quad (3)$$

We use Bayes' theorem to calculate the probability of a player utilizing a specific play style given their play actions. While using play action probabilities only may handicap the performance of the approach, we show below that this approach does converge quite quickly. This approach has proven effective for a perfect information environment in Chapter 3, and we intend to show the effectiveness of its generality over the next few chapters with respect to imperfect information games. Equation (2) gives an extension of Bayes' rule from Chapter 2, Section 2.1 where A is a random variable representing player type, and B is a random variable representing player actions. Our calculation uses an a priori belief of 0.25 as P(A) for each of the four player strategies for the first iteration. The probabilities in Table 4.3 represent an *a priori* P(*B* | *A*) which were obtained by analysing the frequency of past actions of players of style *A* over 100,000 hands (25,000 hands against a collection of each type of opponent). P(*A*) is the prior belief of an opponent's play style, and as such is set to 0.25 initially as all opponent styles are assumed equally likely at the start of a game. Actions that are not influenced by playing style, such as check actions taken when it is not necessary to place any stake to stay in the hand, are disregarded in order to improve the accuracy of the player analysis. Bayes' rule updates the initial probability of player style belief such

that P(*A* | *B*) for the current iteration becomes P(*A*) for the next action analysed (i.e. our belief of style P(*A*) is the result of P(*A* | *B*) for the last action analysed). P(*B*) is represented by the summation of P(*B* | *A*)P(*A*) for all possible *A*, where the updated initial probability represents the probability that the player is of each of those styles. Pseudo code for this analysis player is given in Algorithm 4.1.

```
#BayesTheoremPredictor
Inputs:
LastAction - Opponent's Last action
fLAPlayProb – Array of action probabilities given opponent is LA
fLPPlayProb – Array of action probabilities given opponent is LP
fTAPlayProb – Array of action probabilities given opponent is TA
fTPPlayProb – Array of action probabilities given opponent is TP
for All of Opponent i's past actions
{
  action = Players[i].LastAction;
  // Multiply each action probability by the initial probability
  tp = fTPPlayProb[action] * fInitialTPProb[i];
  ta = fTAPlayProb[action] * fInitialTAProb[i];
  lp = fLPPlayProb[action] * fInitialLPProb[i];
  la = fLAPlayProb[action] * fInitialLAProb[i];
  // Sum all the separate probabilities, to create the normalizing constant,
  // Pr(B), and normalize each value so that the probabilities sum to 1
  float prob = tp+ta+lp+la;
  if (prob < 1)
  {
    tp = tp / prob;
    ta = ta / prob;
    lp = lp / prob;
    la = la / prob;
  }
  // Set all initial probabilities as the new value, Pr(A|B)
  fInitialTPProb[i] = tp;
  fInitialTAProb[i] = ta;
  fInitialLPProb[i] = lp;
  fInitialLAProb[i] = la;
}
Outputs:
fInitialLAProb – An array of LA style belief for each opponent
fInitialLPProb – An array of LP style belief for each opponent
fInitialTAProb – An array of TA style belief for each opponent
fInitialTPProb – An array of TP style belief for each opponent
```

Algorithm 4.1: Pseudocode of the Bayes' Theorem predictor

	Fold	Check/Call	Bet/Raise
LA	0.36	0.05	0.59
LP	0.60	0.29	0.11
TA	0.73	0.02	0.25
TP	0.87	0.07	0.06

Table 4.3: Action probabilities for each opponent style.

If we compare the action probabilities in Table 4.3 to the α and β values from Table 4.1, we can see that the Loose Aggressive player will very rarely check; the narrow interval between LA's α and β (0.1) dictates a very narrow probability for checking. Comparably, the very large probability of betting is indicative of the very low threshold for betting behaviour. The fold probability appears to be quite large for the LA player, but it must be considered that the analysis is upon a subset of all of the players actions (due to the aforementioned disregarding of certain betting scenarios) and therefore the probabilities are skewed. Similarly, the checking probability over all types of opponent are quite low due to this situational ignorance except in the case of the Loose Passive player, which will check and call significantly more than the other styles due to the greater number of hands played given the low α value. As is atypical of the tight player, fold probabilities are greatly exaggerated the tighter the player becomes, and it can be observed that the TP player will both check and bet very rarely given the high values of both α and β.

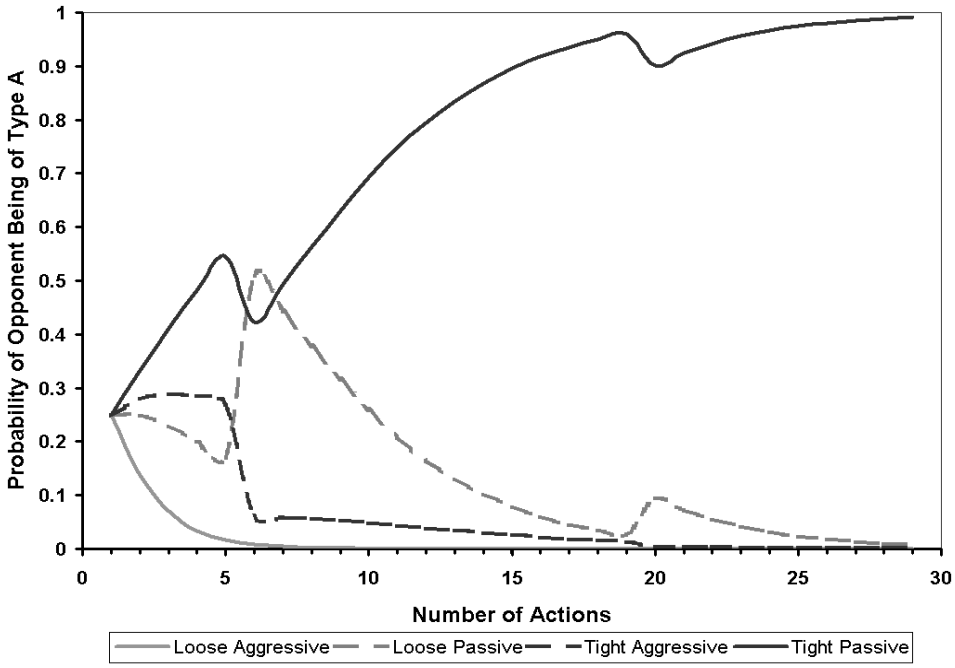


Figure 4.2 : Graph displaying how player type probabilities alter given opponent actions. In this case, the analysis player converges to the conclusion that the opponent is a Tight Passive player

Figure 4.2 gives an example that shows how quickly the analysis player probabilities converge. This exemplifies the speed at which Bayes' Rule can determine player strategy, as long as we have the probabilities of low level actions over some well-defined player strategies (or potentially intentions/goals). This approach could have a wide application over a number of games where targets and intentions need to be modelled, especially where a representative set of possible strategies can be found, possibly even utilising network data or play records within a game - this is discussed further in Chapters 3, 5 and 6. The heterogeneity of independent strategies is vital to the accuracy of the analysis; in this case, we have a somewhat distinct separation between the strategies in relation to action probabilities (a conditional independence) that can be seen in Table 4.3. For example, the TP player can be defined by the frequency of folding, whereas the high frequency of folding with the TA player can be

offset from TP analysis by the (comparatively) significantly greater betting frequency. In the same vein, the TA player is separated from the LA player by the greater folding probability. The risk of strategy similarity can result in misleading analysis, as can be the case with scenario disparity in comparison to playing style. To explain this, consider a situation where an LA player holds a relatively weak card (5, for example); in this scenario, the LA player will bet significantly, whereas a Tight Passive player will most likely not stay in the hand given this dealing. If we compare this to the scenario where a TP player holds the highest card in the deck (an Ace), the TP player will also bet profusely. In both of these situations (if given no prior analysis) the Bayesian approach would assume the style for both players (a Loose Aggressive one) given the intensity of betting actions. Subsequent hands will separate the two players in terms of style to the analysis, but the similarity of play given certain scenarios between two differing styles can occasionally lead to an erroneous analysis. There must be a significant difference between the action probabilities of style (or goal in the case of Chapter 3) for a good enough conclusion to be reached. A good example of this can be seen in Chapter 3, where Figure 3.4 displays indecision as to the exact goal of the opponent given the (initially) similar behaviour when attempting to reach on of two disparate locations.

4.2.5 Modeling Correct Responses for Differing Styles

The four Anti-Players combine to form the strategies of the Analysis Anti-Player. After the analysis player learns the opponents' styles, it prioritises its reactions in relation to the most "dangerous" opponent type. For example, a Tight Passive player's

actions would be taken more seriously than that of a Loose Aggressive player. This risk analysis leads to choosing a specific anti-player's tactics dependent on the greatest threat. A Tight Passive player has the greatest priority, due to the tight risk-free nature of play. After this, a Tight Aggressive player would be given priority.

When considering only loose style players, the most frequent style is given precedence (for example, in a set of two LA players and a single LP player, LA is given precedence), pigeonholing the entire set of opponents in relation to that specific style. Algorithm 4.2 shows how the Analysis Anti-Player chooses its tactic. The player analyses each of the opponents' actions and determines the opponents' play style using the algorithm defined in Algorithm 4.1. It then analyses the assumed styles of all the opponents, and chooses to play against the most threatening style; Tight Passive and aggressive players take priority as tight players are much stricter in their style of play, if no tight players exist, then priority is given to the loose style used by most players. When the Analysis Player has decided which style of player to respond to, it will use the relevant 'Anti' style of play. This importance is in line with the best exhibited performing players in Kendall and Willdig's work, where a tight approach to play is generally preferred (Kendall & Willdig, 2001).

```
#AnalysisAntiPlayer
Inputs:
NumberOfTP – Number of Tight Passive Opponents
NumberOfTA – Number of Tight Aggressive Opponents
NumberOfLP – Number of Loose Passive Opponents
NumberOfLA – Number of Loose Aggressive Opponents
BayesTheoremPredictor()
CountNumberOfOpponentStyles()
if(NumberOfTP>0)
{
   action = GetAntiTPActions();
}
else if(NumberOfTA>0)
{
   action = GetAntiTAActions();
}
else if(NumberOfLP>NumberOfLA)
{
   action = GetAntiLPActions();
}
else
{
   action = GetAntiLAActions();
}
Outputs:
action – The action to be performed by the player
```

Algorithm 4.2: Pseudocode for the Analysis Anti-Player to determine the opponents' style and choose which tactic to employ

4.2.6 The Simulation Player

The last player created is called the Simulation Player. This player, on each of its turns is told the style of each opponent, and it then runs a simulation of the game within itself with the current demographic of players, and runs tests against the players over a discrete set of 66 α and β values in increments of 0.1 where $0 \leq \alpha \leq \beta \leq 1$. These tests consist of 100 games per [α, β] pair. This process takes approximately 5 minutes. The player then utilizes what it sees as the 'best' values of [α, β] to make subsequent decisions against the opponents. The graphs from Figure 4.4 to Figure 4.7 show a representation of the results that a simulation player receives when playing different sets of opponents.

4.3 Results and Analysis

4.3.1 Bayesian Analysis vs. Simulation Player

The comparison of the Analysis Anti-Player against the Simulation Player (Figure 4.3) represents an average of the percentage of tournaments won by each player against every combination of opponents in a four-player environment. All experiments are run on a Pentium IV 3.0 GHz HT with 1GB RAM using C#.NET running under Windows XP SP2, with a computation time of 50 minutes.

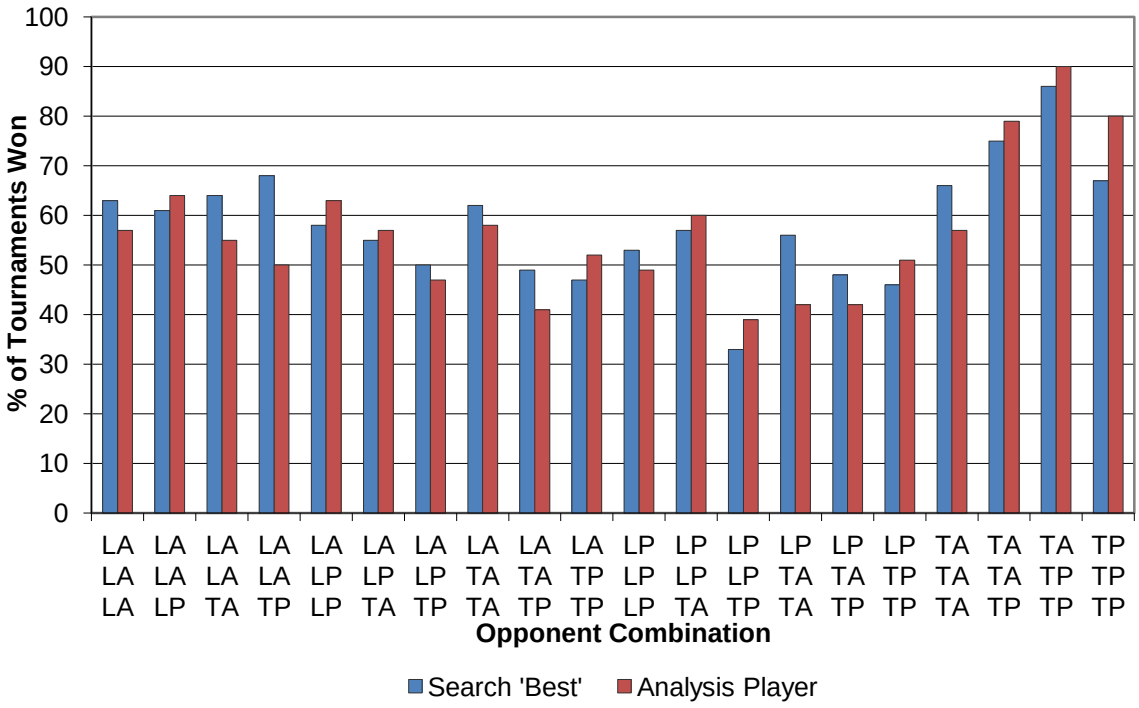


Figure 4.3: Graph displaying combinations of opponent styles against the percentage of tournaments won by each player

We observe that the averaged results are not sensitive to the order of opponents around the table, with a confidence interval of 3%. We can see that the performance of the Analysis Anti-Player is comparable to, and on occasion surpasses that of a player

that is already knowledgeable of the opponent's styles. Due to the random nature of the hands, it is not surprising that the Analysis Anti-Player is occasionally the best. These results show how successful the pigeonholing technique is compared to a 'custom built' design that the Simulation Player creates for the current demographic even though the analysis player is over 100 times slower (due to the large number of simulations needed at each decision point).. There is a point where the success rate of the Simulation Player reaches below 40% (the analysis player is also significantly impaired). The opponents at this point consists of two LP and one TP player – we consider that the loose nature of the two LP players influenced the simulation player to stay in more hands given the passive nature of the opponents, but the non-adaptive nature of the player does not adjust for when it is facing only the TP player (when either the LP players have either not played in the hand or have been previously knocked out of the tournament). The greater performance of the Bayesian analysis player can be attributed to the dynamic adjustment it performs given the changing opponent demographic, although it still displays difficulty due to the precedence of the TP players presence which dictates that more hands should be stayed in (which somewhat counteracts the advice of infrequently staying in the hand against a looser player) It should be appreciated however, that 40% is not necessarily a failure as this is a four-player game, and 40% is greater than a 'fair share' of tournament wins.

One comparison between the two players is related to the performance of each player against the groups of mostly tight players on the right hand side of the graph.

These results show that the players are both quite competent against these styles of play, and the Analysis Player has nearly an 85% success rate. An explanation for the high rate of success, even against four players is possibly due to the tight nature of the opponents; any action that the player takes which is not a folding action may steal the pot from a tight player with a better hand: tight players are very susceptible to bluffing.

The success against mostly loose players is slightly less prominent, but still very impressive, averaging around 55% of tournament wins. This is not surprising since a loose player's actions reveal relatively little about the card held. This can lead to situations where a loose opponent will have a very strong hand, and the Analysis Players will still remain in the hand, resulting in an unsuccessful tournament.

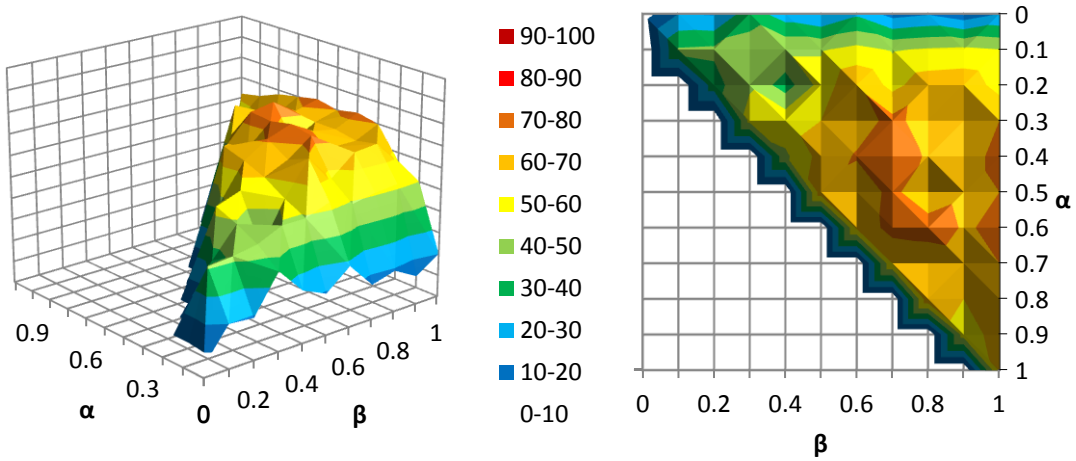


Figure 4.4: Graph displaying the tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LA/LA opponents

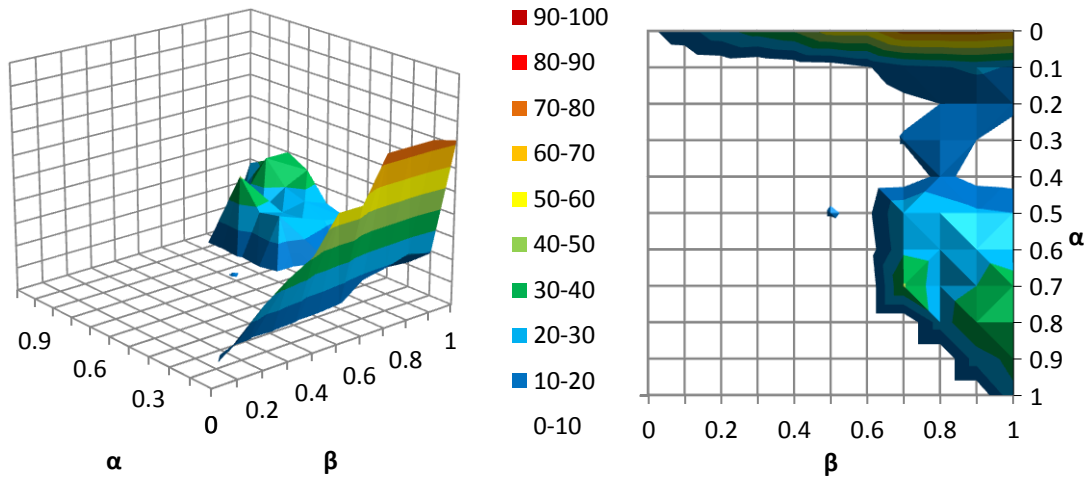


Figure 4.5: Graph displaying the tournament success of different values of α and β when playing a four-player tournament against TP/TP/TP opponents

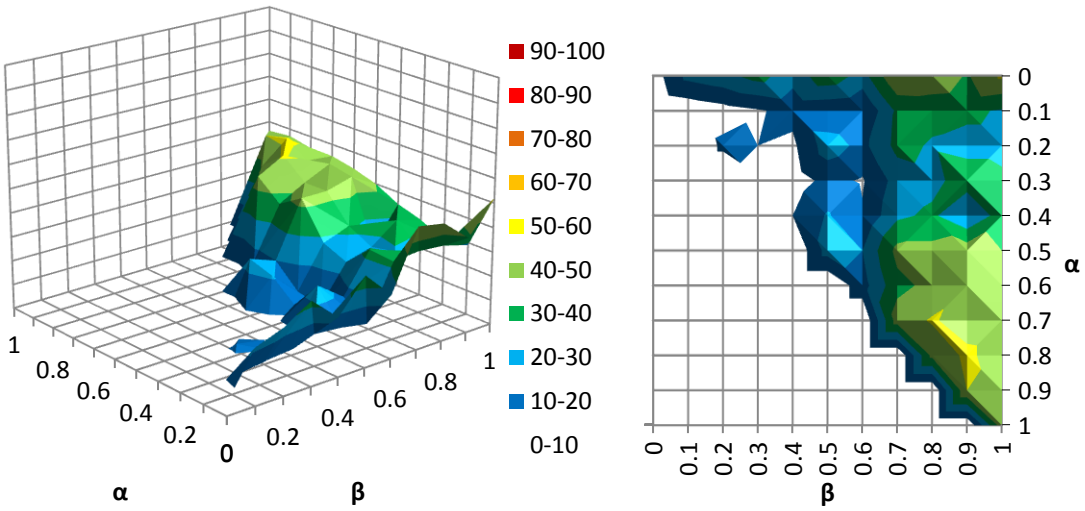


Figure 4.6: Graph displaying the tournament success of different values of α and β when playing a four-player tournament against LA/LA/TP opponents.

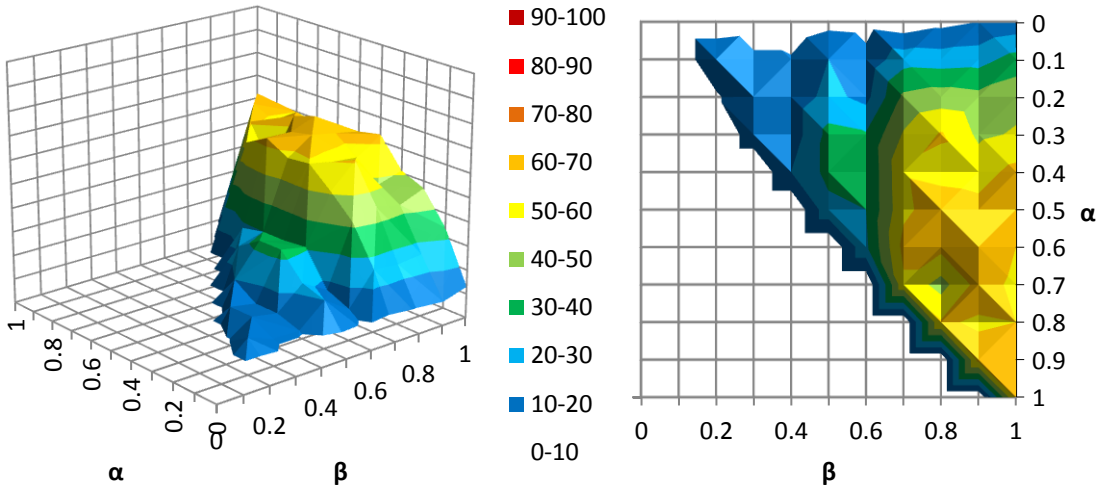


Figure 4.7: Graph displaying the tournament success of different values of α and β when playing a four-player tournament against LA/LA/LP opponents.

4.3.2 Performance against Static Styles

Figure 4.4, Figure 4.5, and Figure 4.6 provide some explanation as to why the pigeonholing technique works; these figures show the success of each possible value of α and β against a different collection of opponents; the areas closer to red indicate a greater concentration of large win percentages. The entire collection of results for each combination set of 3 opponents is in Appendix A.

Figure 4.4 shows the successful probabilities against three Loose Aggressive opponents, a large area of high success values can be seen, indicating that playing tightly, as well as playing with a low amount of belief in the severity of the opponent's actions can bring a large number of wins. Figure 4.5 shows the successful probabilities against three TP opponents, and the graph shows that the concentration of wins is very low, mainly due to the low probability of a large pot when playing a tight player, especially when playing against a Tight Passive player. The graph indicates that the

best strategy is to participate in as many hands as possible, but not to make a betting action unless completely certain of a win. This appears to be a rational tactic, as remaining in a hand increases the probability of stealing a pot from a tight player.

Figure 4.6 shows the effect of replacing one Loose Aggressive opponent with a Tight Passive one. The transformation from Figure 4.4 to Figure 4.6 is quite dramatic, and exemplifies the effect that a Tight Passive player can have upon a game of three loose opponents, in turn justifying the pigeonholing technique, as the resulting graph is much more closely related to the graph of three Tight Passive opponents than it is for three Loose Aggressive opponents. The main explanation of this is probably due to the order that a game may take with this general demographic; the loose players would have a tendency to risk many chips, and may get removed from the tournament early, leaving only the Anti-Player and the Tight Passive opponent, which would probably dominate most of the games. It should also be noted that the number of tournament wins in total across the graph is relatively low compared to other graphs. This is however, one of the points where the Analysis Player greatly outperforms the Simulation Player (by nearly a 20% margin). This is arguably due to the Analysis Player modifying its behaviour while playing to deal with the greatest threat. When the loose players have been removed, the main focus of play will be against the Tight Passive player.

Figure 4.7 shows how small an effect is brought upon the same demographic of Loose Aggressive players by adding a Loose Passive player. As can be seen, there is very little

difference between the graph in Figure 4.7 and that of Figure 4.4, which is mainly caused by the loose nature of the opponents, as the dark area of the graph (displaying an 'area of believability') shows how fragile a loose player is against partially tight play.

4.4 Summary

This chapter presents an adaptation of the Bayesian analysis introduced in Chapter 3 for a simplified game of Poker. The analysis is implanted into an agent that analyses the past actions of its opponents, and in turn uses the Bayesian analysis to learn the style of opponents it is facing. The player uses a system of ranking to determine the greatest threat and acts as if all opponents are of that style by 'pigeonholing'. The performance of the learning player compared to one that knows the opponent styles without the need for probabilistic calculation is quite similar, and demonstrates the effectiveness of both learning and pigeonholing.

A single deterministic opponent proves simple to beat once the Bayesian predictor analyses its actions and determines the opponent's style. When facing a group of three opponents, however, the pigeonholing technique works to a suitable degree, but could be compromised when coping with partially randomised, dynamic, or bluffing players. An expansion to this technique would require a design highly specific to the strategies which the opponents might have could prove highly unwieldy. Given this limitation, we shall consider an alternative means of agent construction in the next chapter.

Chapter 5: Evolutionary Approaches to Simplified Poker

Chapter 4 notes the difficulty of manually constructing a decision-making agent, such that a hard-coded agent may be limited by its construction in relation to achieving a successful counter strategy to the current opponent demographic. In this chapter we evolve a population of agents using a genetic algorithm in order to replace the previous 'anti' players. We further investigate the ability of a single chromosome to play against all four types of opponent then investigate the effects of an opponent model upon the quality of the evolution in order to ascertain that any improvement can be gained through the use of an opponent model with respect to the evolved solutions employed. We further apply this investigation to a neuroevolutionary approach which has been shown as a suitable general-purpose means of providing an effective agent (Lockett & Miikkulainen, 2008). We investigate the ability of NEAT to construct reliable counter strategies to our initial static opponent styles, and subsequently the effect our opponent model has upon the effectiveness of the evolved player, with the intention that an evolved player can be able to develop more complex strategic behaviours given our Bayesian modeling technique. Using our Bayesian approach, this chapter demonstrates the importance of recurrence and opponent modeling in conjunction with neuroevolution when investigating multiplayer games. Finally, the robust nature of our model and evolved agents is evaluated against opponents that use varying levels of bluffing and adaptation of their play style.

5.1 Evolving Genetically-Coded 'Anti'-Players

We evolve a set of players using an evolutionary algorithm using a chromosome comprised of 60 genes. The chromosome is built from three sections, based upon the opponent's most recent action. In this case, a bet or a checking action is considered, but a folding action is not (as it indicates the end of a hand in our 'heads-up' environment) – we do, however, use the third section for the situation in which we are currently playing a new hand. In each of these three sections, the 20 genes determine both the alpha and beta values for each potential card held. For example, if the first two values of the chromosome are 0.5 and 0.7 respectively, then these values represent that the opponent has made a bet, stay in the hand with probability 0.5, and bet with probability 0.7 if staying in the hand. The alpha-beta pair is not separated during crossover, as keeping the two values together reduces noise (although both values are individually subject to mutation). This design is shown in Figure 5.1

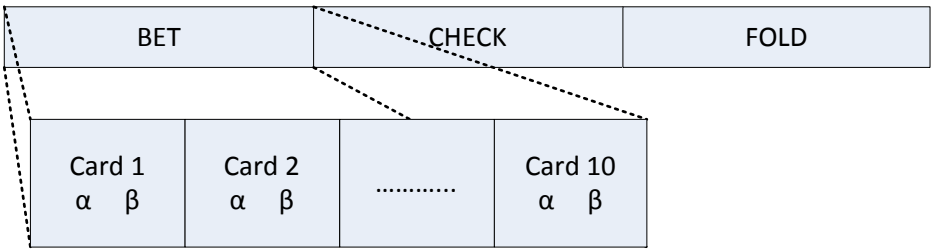


Figure 5.1: Design of the Chromosome

The fitness of the solutions is determined by their performance in a 100-game scenario where performance is ranked by the number of games won by the solution. In this case a 'game' represents a competition where the winner is the player that wins all of the chips from his opponents. At the start of the game a player receives 100

chips, places an ante of 5 chips, with bets limited to 5 chips per action and limit of 40 chips per player per betting round (essentially four bets, per player, per round). Once the bet limit is reached, any agent that would generally choose a betting action will call the current maximum bet instead of re-raising. Solutions are evolved using a population size of 100, a single-point mutation probability of 0.01 (intending for one mutation per chromosome, per generation), and single-point crossover for 500 generations. We also use a fitness proportionate (roulette wheel) operator to promote poorly performing, yet potentially useful solutions to breed with other successful solutions. Each experiment is run five times and the results shown represent an average of those five runs.

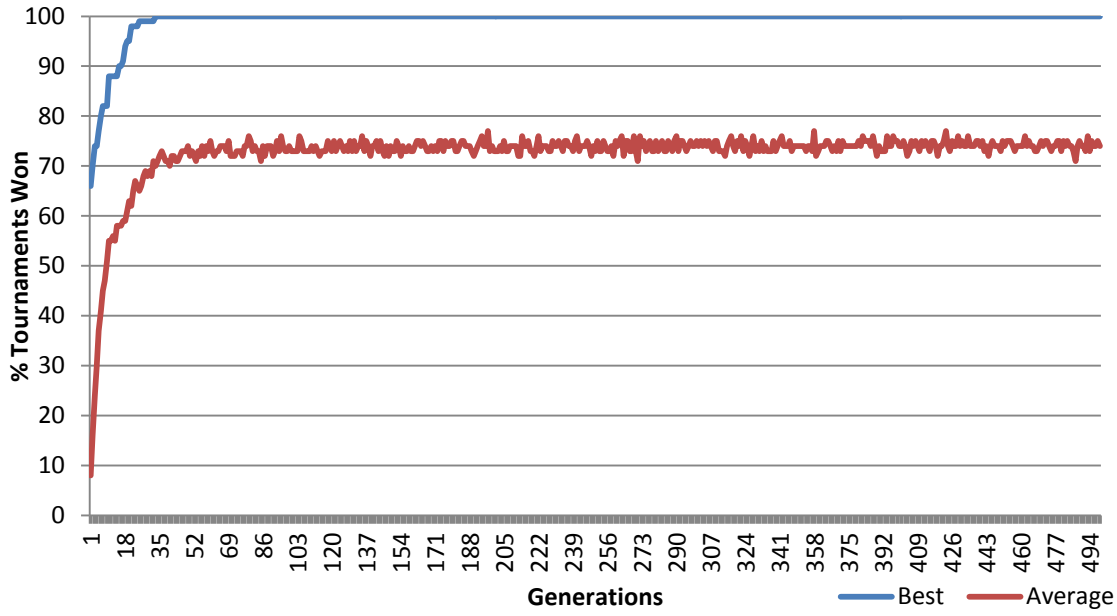


Figure 5.2: Evolution of a GA player against a single Loose Aggressive Opponent

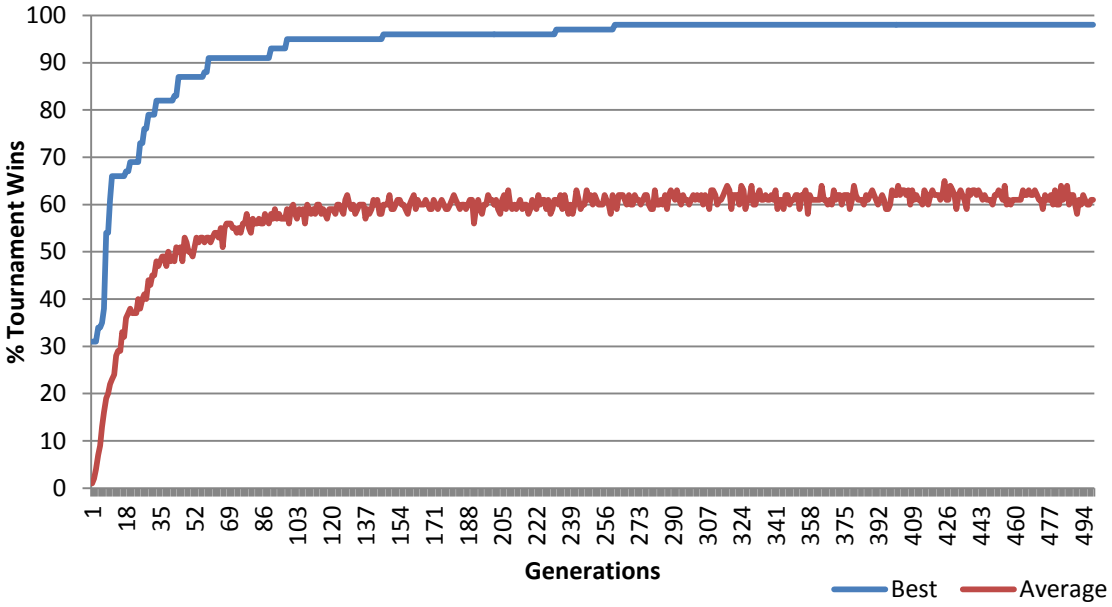


Figure 5.3: Evolution of a GA player against a single Loose Passive Opponent

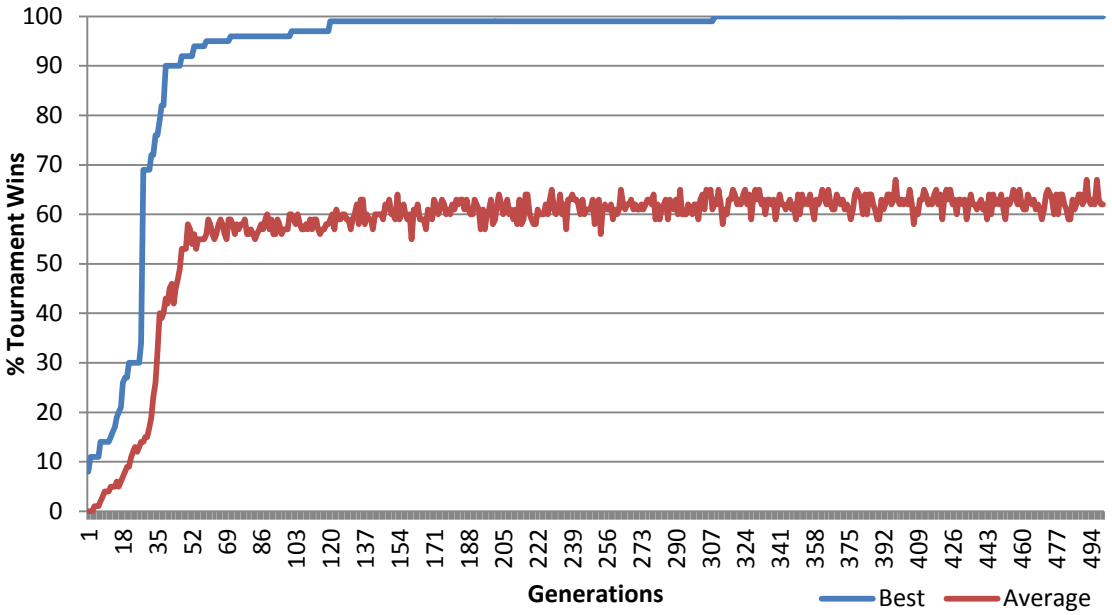


Figure 5.4: Evolution of a GA player against a single Tight Aggressive Opponent

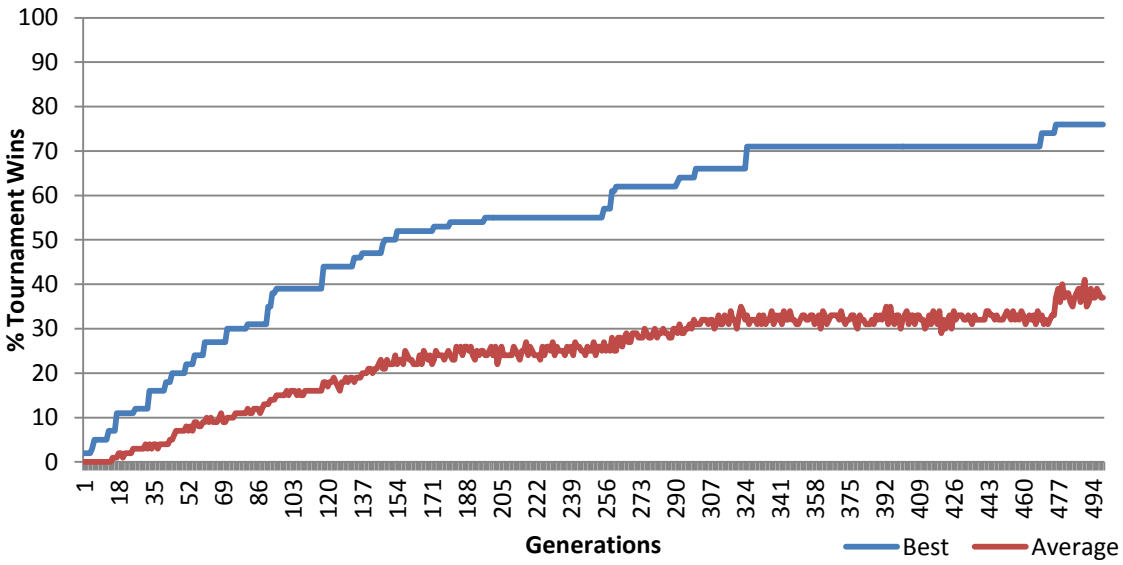


Figure 5.5: Evolution of a GA player against a single Tight Passive Opponent

The evolution shown in the graphs from Figure 5.2 through to Figure 5.5 show that an evolutionary approach to defeating each player is accomplishable, as has been shown in other work (Barone & While, 2000) (Kendall & Willdig, 2001). Particularly in Figure 5.2, we see that the average 'best' solution for the first generation has an extremely high fitness (with almost 70% of tournaments won) which can be attributed to the Loose Aggressive player performing mostly betting actions. This behaviour results in mainly the first 20 values of the chromosome being utilised (those within which responses to betting actions are performed). As with the nature of the randomly initialized population, the α and β values of the 'best' overall solution from the first generation indicates a probability of staying hands of around 0.5 (with a card of 4 or greater), which is analogous to the behaviour indicated by the Anti-LA player from Table 4.2. These values are evolved over (an average of) the next 40 generations to procure a 100% win record against a Loose Aggressive player. The evolution of an Anti-

Tight Aggressive player seen in Figure 5.4 is similarly successful after an understandably longer period of evolution, as the Tight Aggressive player is more likely to be staying in hands with a stronger card rather than the frequent Betting of the Loose Aggressive player. Looser players appear to be a more difficult task for the evolution to compete against – the evolution against the Loose Passive player in Figure 5.3 displays a similarly fast initial evolution, but the evolution slows dramatically, showing an evident stagnation in the average population fitness from around generation 120, the average best performance reaching a win percentage of 98% over all five instances of evolution. We see further difficulty with the evolution against the Tight Passive player (Figure 5.5), which upon reaching the limit of 500 generations still appears to be evolving (as indicated by the significant jump in average population fitness around generation 470). The slow evolution is understandable due to the difficulty previously discussed about the strength of the TP player in Chapter 4, section 4.3.1, but even at the end of the 500 generations, the average best performance of the evolved solution is of 76% tournament wins, which is only of 1% different to that 'Anti-TP' player displayed in Table 4.2. The evolved Anti-TP player behaves similarly with respect to the two-parameter 'Anti-TP' player, although a more diverse play behaviour appears such that folding is more likely in response to a Tight Passive opponent betting, and a persistence to stay in most hands where a Tight Passive player is likely to fold.

5.1.1 Evolving against all Types of Opponent

The ease of our evolved agents to generate a successful counter strategy begs an investigation into determining whether a single chromosome could evolve a strategy capable of competing adequately with all types of opponent. We once again evolve solutions with the parameters given above, but run the experiment for 1000 generations. Although the evolution shown in the graphs from Figure 5.2 through to Figure 5.4 show relatively rapid evolution to strong solutions (approximately 400 generations), the slow evolution shown in Figure 5.5 dictates that a greater amount of time for evolution be provided given the increased complexity of the problem attempted. The fitness now consists of an average of how the solution performs when playing 100 games against each style of opponent in turn (hence 400 games per solution evaluation, 100 each against LA, LP, TA and TP), with a computation time of approximately 8 hours per opponent. The results of this evolution are averaged over five individual runs, and are displayed in Figure 5.6

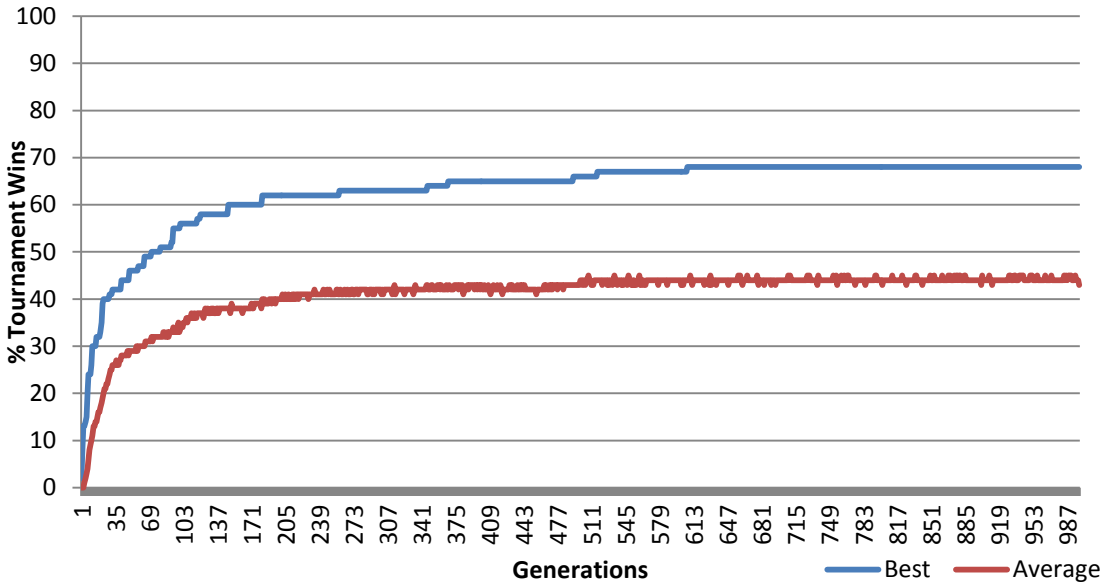


Figure 5.6: Evolution of a GA player against all types of opponent

Figure 5.6 shows that the collective average is quite poor in relation to the potential performance shown from Figure 5.2 through to Figure 5.5, although a 'best' score of 68% against all four opponents is not necessarily a bad average, given that a successful player will win more than 50% of the time. Through observation of the overall 'best' evolved solution, we can see that the performance against both styles of Aggressive player is very strong (with 100% of tournament wins against each), the overall solution is weak in comparison when playing against the Loose Passive and Tight Passive Players (with average Tournament successes of 40% and 32% respectively). Using a single chromosome in order to respond to all four style of opponent is a weakened approach to facing the various types of opponent. We believe that utilising our Bayesian opponent model as part of the evolution could help yield stronger solutions against all four types of opponent. We now evolve solutions using a chromosome length of 240 (four sets of 60 genes as represented by our earlier experiments), and

use our Bayesian opponent analysis to determine which part of the chromosome to use when making decisions of when to stay in and fold, such that the first 60 alleles are used as a response to the belief that the opponent is Loose Aggressive, the second 60 to respond to a Loose Passive player, and so on. The averaged evolution (over five instances) of our chromosome using Bayesian selection is displayed in Figure 5.7.

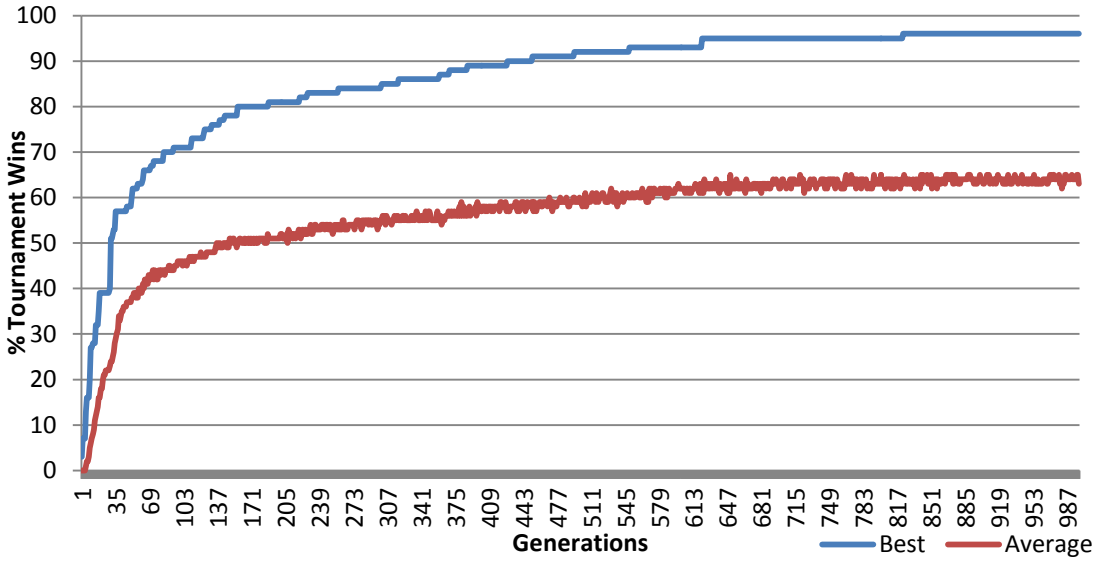


Figure 5.7: Evolution of a GA player using the Bayesian Opponent Model against all opponent styles

Figure 5.7 shows the evolution of our agent using our Bayesian analysis, which is, in comparison to Figure 5.6, far stronger when facing all types of opponent. The dramatic improvement is accountable to the separate sections of the chromosome being allocated to each opponent style, this allows flexibility enough that separate counter styles (like those evolved in Figure 5.2 through to Figure 5.5) can be utilized specifically for each opponent such that once our Bayesian predictor assumes the opponent style to be Loose Aggressive, for example, all resulting actions are taken from the first 60 alleles in the chromosome (until the model adjusts, of course). The rapid evolution of

the solutions is surprising in comparison to Figure 5.6, where Figure 5.7's chromosome is four times larger in its construction. The ability for Bayes' Rule to select between individual strategies within an evolved agent is an important ability, slighted only by the need for separate tactics to be evolved (a per style set of genes within the chromosome). The resulting chromosome is, as mentioned four times larger than the other solutions evolved, which is analogous to Bayes' Rule selecting one of the four agents evolved in Figure 5.2 through Figure 5.5 (similar to the selection of an Anti-Player in Section 4.2.4) instead of a single, smaller solution. Furthermore, this genetic approach may still be constrictive in approaching stronger solutions against individual styles (with particular reference to the difficult evolution against a Tight Passive player shown in Figure 5.4).

5.2 Evolving an ANN-Controlled Opponent

In order for us to investigate an agent's ability to evolve more complex behaviours, we take a neuroevolutionary approach to our simple form of Poker. The capability of evolutionary neural networks to adapt to their environment, as well as changes in the environment itself is a tributary factor to our choice of this approach. The adaptive nature of a neuroevolutionary system enables the determination of behaviour (and potential performance benefits) of an agent with which the internal decision process cannot be directly controlled or defined by the author, which can in many cases be a restrictive factor in agent design (Yao & Liu, 1998) (Yao, 1999).

5.2.1 SharpNEAT

For our agent evolution, we use a C# version of Stanley's NEAT algorithm called SharpNEAT. NEAT evolves increasingly complex networks, meaning that it becomes more complicated as evolution progresses, potentially meaning that more complex and sophisticated behaviours can be evolved (Stanley & Miikkulainen, 2002) (Stanley & Miikkulainen, 2002) (Stanley & Miikkulainen, 2002). An evolutionary algorithm is used, which utilizes historical markings, separating innovations into separate species, and gradually increases the size of the networks involved, determining the structure of the ANN. Each connection gene stores the input and output nodes, as well as the connection weight and, if it is enabled, an 'innovation number' which helps find corresponding genes during genetic crossover.

Mutation in NEAT affects nodes, weights and connections, with nodes being added (splitting a connection, disabling one connection and creating two more), or new connections between nodes being created. Crossover involves matching genes between two equal-fitness parents, and using disjoint and excess genes to create an expanded child node. The population is speciated so that innovations are not lost; only new innovations (with same innovation number) are compared to each other, rather than the entire population. The fitness of a single *genotype* is the determining factor in the existence of the species, which is based upon the averaged fitness of all genotypes within that species. This restriction means that greater populations are at greater risk of more adaptation or replacement.

5.2.2 Evolving NEAT to Play Poker

An issue for every researcher using ANN's in their work is that of the number and form of their inputs, Davidson et al. (2000) used an ANN to predict an opponent's next action, and used binary values for Boolean inputs representing the stage of the game and the last action of an opponent, and real values from 0 to 1 for all others (such as pot odds) in order to represent opponent playing habits. The design of our network structure, which ignores the round-based and card-based information from Texas Hold'em used in Davidson's approach, can be seen in Figure 5.8.

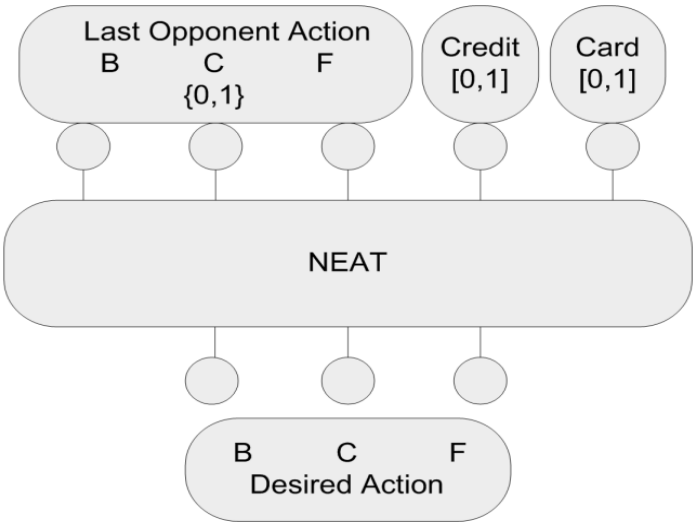


Figure 5.8: Design of player network.

The 'Last Opponent Action' inputs translate the last action of the opponent into binary. The 'Current Credit' input represents the ratio of chips the player has in comparison to the number of chips available at the table (including all opponent-held chips). The final (and arguably most important) input is that of the card held, which is represented by its number divided by the total number of cards. The outputs of the network represent the action to be taken, the action represented by the largest output

value is the action performed.

In our experiments the fitness of each of the genomes in a population is represented by *T*, the percentage of tournaments won by the player (i.e. the tournaments where this player wins all the chips of all opponents) over 100 tournaments. We use the number of tournaments won due to the speed with which an iteration of a tournament of one card Poker can be run (in comparison to that of Full-scale Texas Hold'em); in this case success represents a genuine 'competition' win rather than a potentially erroneous success (i.e. winning a series of big pots through luck before a hand limit is imposed, for example). We use a population size of 100, a node-addition probability of 0.005, and a node-connection addition probability of 0.01. The new node addition probability is limited such that several generations may be required to adjust to the addition of the new node (and for any connections added between the new node and others). Similarly, too great an addition frequency may reduce the effectiveness of the network in making effective use of the nodes currently in the network. The node connection addition probability is such that in a single generation there will be (assuming the addition of more nodes to the eight initial nodes per network) around one connection added per network (Stanley et al., 2005). Note that these results have been run for 400 generations (with a computational time of approximately 30 hours per run), but graphs have been truncated in order for ease of reading, and has been done so only when no further improvement has been exhibited by the evolution. It should be further noted that, with respect to the 'best' solution at the end of

evolution, the results given against LA and LP opponents have a confidence interval of 2 tournament wins (2% of all tournaments played in this case) from the mean, and those against TA and TP opponents have a deviation of 5 tournament wins (5%). This is due to the stochastic nature of the cards in determining success, as well as the difficulty of playing against tighter styles of player.

As we can see from Figure 5.9 to Figure 5.13 neuroevolution works very well versus a static opponent, with results against LA, LP, and TA styles reaching a 'best' success rate of 100%, with average population fitness above 90%. This is unsurprising, as even a static counter-approach to defeating static styles can be successful as shown in the previous chapter, and here we are using a dynamically learned approach.

Through analysis of the outputs we have seen that, after evolution, the values output by the network generally give a very clear decision of the action it wishes to perform. The network structures in this case are somewhat simple; an evolved link between the card strength input to one of the outputs generally causes the dramatic increase in fitness in most cases (generation 56 in Figure 5.9; generation 23 in Figure 5.10; generation 20 in Figure 5.11). The results of evolution shown are an average of five individual SharpNEAT runs, the computation time of each run is approximately 4 hours.

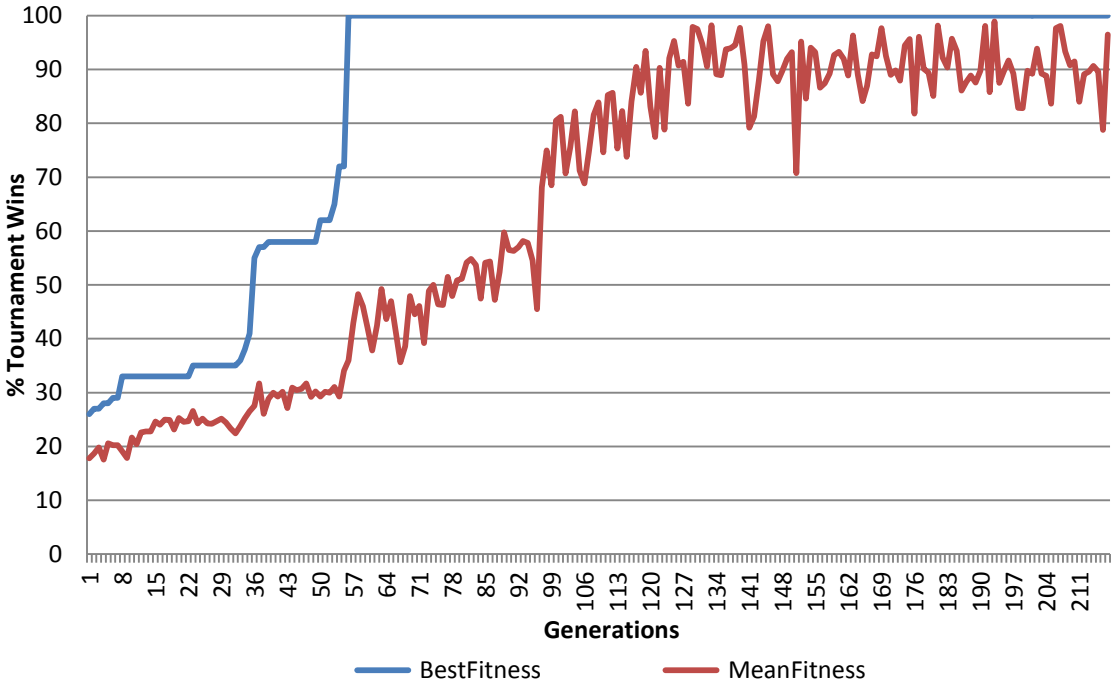


Figure 5.9: Evolution of an ANN against a simple Loose Aggressive Player.

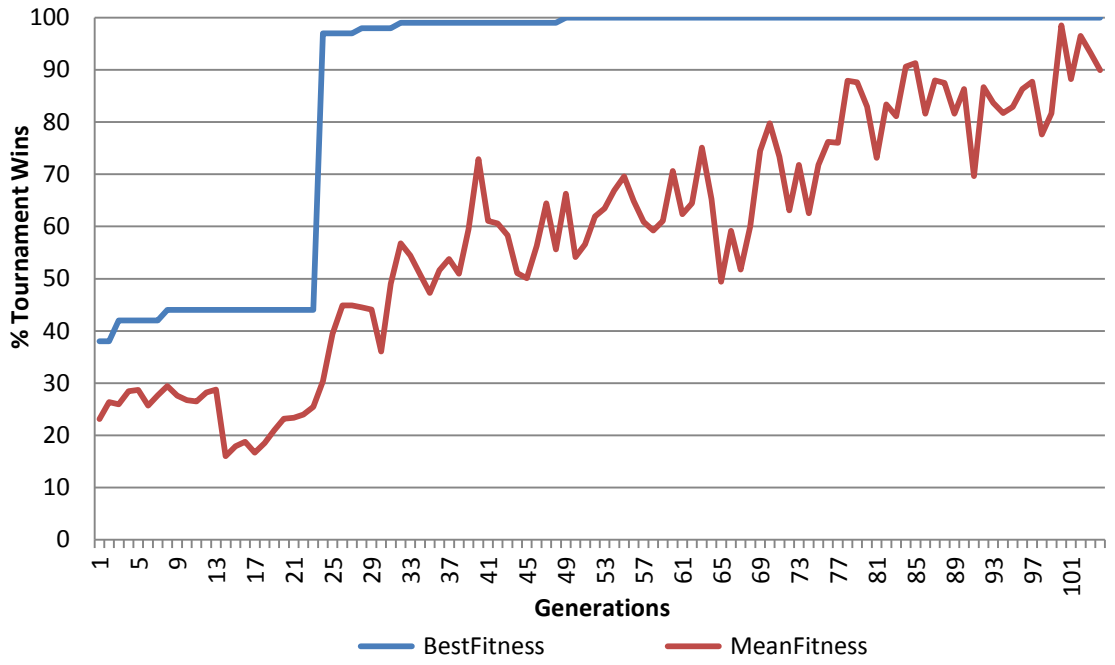


Figure 5.10: Evolution of an ANN against a simple Loose Passive Player

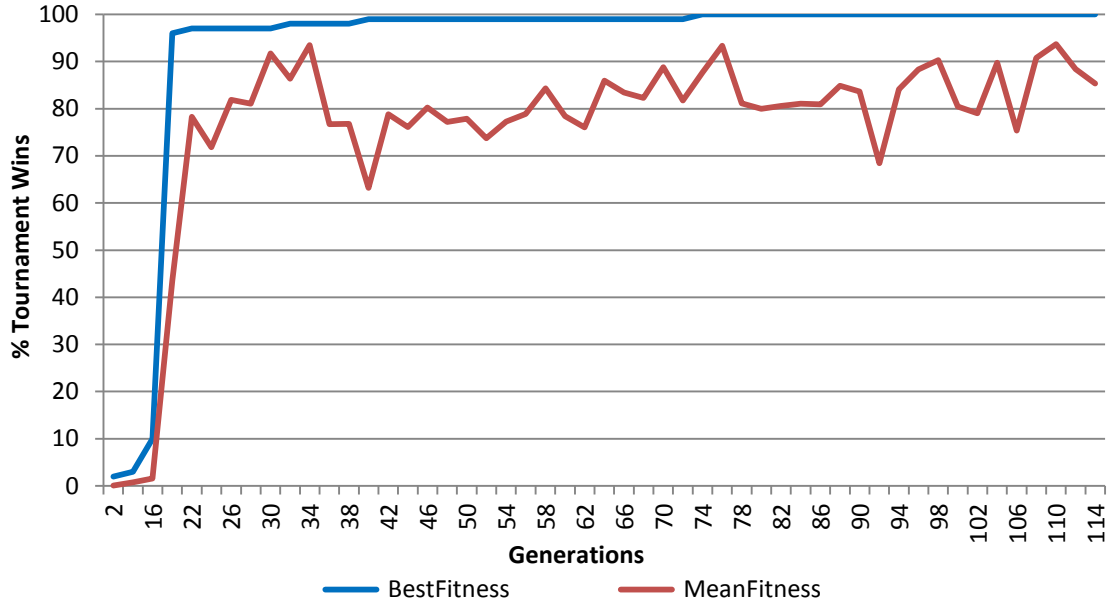


Figure 5.11: Evolution of an ANN against a simple Tight Aggressive Player

In the evolved best network there is a definite trend towards avoiding linking card strength to the bet output with a positive weighting; most connections to a betting output come from inputs/nodes linked to the opponent's previous action. Another frequent reoccurrence of note is that of a positively-weighted recurrent connection to, and from the betting output node (i.e. a node that links to itself): it is fair to believe that if the previous action is a betting action, the recurrence in this node would yield that the agent should keep betting upon requesting the agent's next response. This occurrence can be observed in Figure 5.12.

When we observe Figure 5.13, however, we notice that our evolution fails to reach a 100% success rate against a single Tight Passive player, even after many generations. This is due to the Tight Passive strategy being stronger than the other static strategies, and also the hardest to gauge due to the frequency at which checking actions occur

over any display of strength. Nonetheless, even in this case tournament success of the evolved NEAT network is impressive.

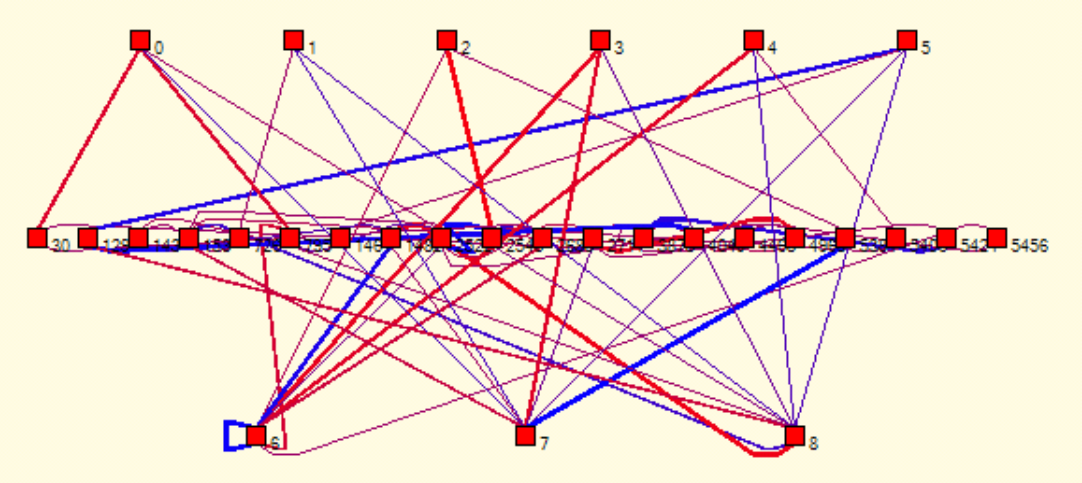


Figure 5.12: Network Structure of the evolved best network from The evolution shown in Figure 5.11

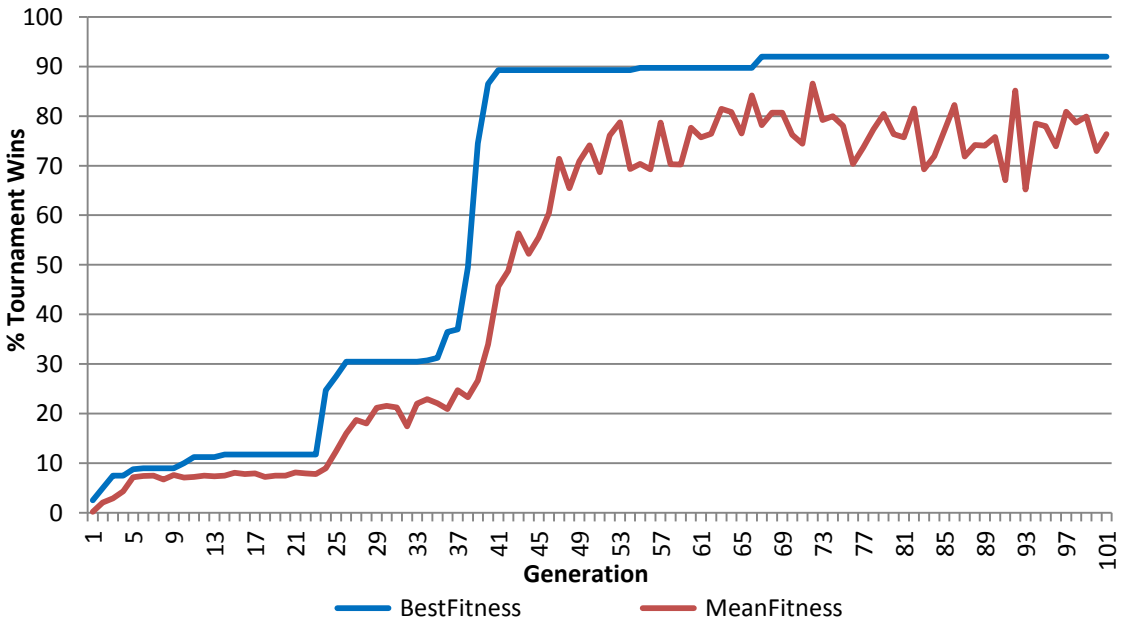


Figure 5.13: Evolution of an ANN against a simple Tight Passive Player

Having now shown that a NEAT-evolved player is capable of defeating a single individual style, we aim to evolve a player capable of defeating all four types of opponent; Figure 5.14 shows the evolution of a player using the inputs described in

Figure 5.8. We make each candidate solution play 100 games against each opponent in every generation, such that they will play 100 games against an LA opponent, then 100 games against an LP opponent, and so on – the order in which opponents are played is random, and the potential for our recurrent ANN to simply "remember" the player type from one game to the next is removed through the clearing of all data within a network between game instances (i.e. the data that is within the recurrent connections between nodes – the structure of the network and its weights remain) (Stanley & Miikkulainen, 2002). The fitness *f* of each solution is again the percentage of tournaments won against all opponent styles. It is noted that the evolved player reaches a maximum average success rate of 87% over 100 tournaments. If we consider Figure 5.9 through Figure 5.13, we can tell that our players should potentially be able to gain a greater success rate than this against these opponents. The much increased difficulty of the task of beating all four player styles is demonstrated by deeper analysis of the network structure – which is extremely complicated, and full of recurrent connections. The number of hidden nodes in the resulting evolved players is more than double that of those evolved against the individual styles from Figure 5.9 through Figure 5.13. It could be posed that our evolving agent could have been attempting to (and potentially succeeding in) constructing a means of modeling the opponent. At generation 136 in Figure 5.14, we can see a significant rise in the fitness of this agent, network analysis shows that a connection from a 'mess' of network connections to the output node for a checking action gave the boost required by the network for the increased fitness. In general, however, completely understanding the effect on game

play of the connection topology and weights is extremely difficult; Towell and Shavlik describe the ANN as a 'black box' , and that 'determining exactly *why* an ANN makes a particular decision is all but impossible' (Towell & Shavlik, 1994).

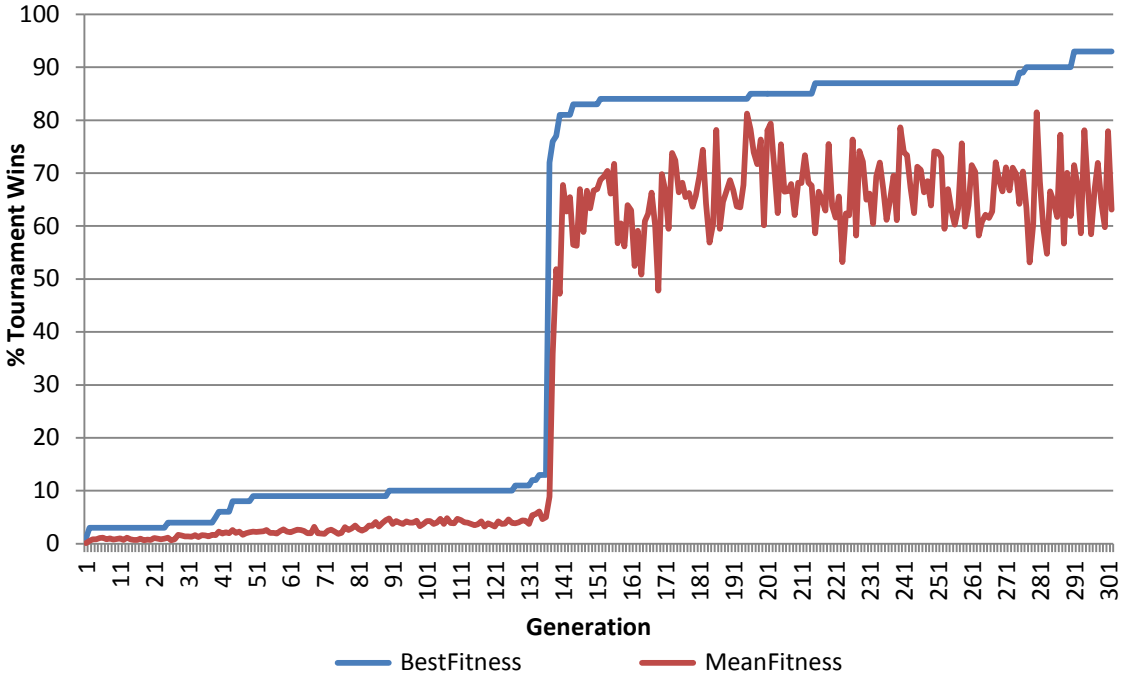


Figure 5.14: Evolution of an ANN against all types of opponent

5.2.3 Augmented Evolution with Bayes' Rule

An opponent model can represent the individual nature of an adversary, and as such could aid in the correct selection of an appropriate reaction to each opponent by an evolved network. In preparation for experimentation, we conclude that the Bayesian beliefs of opponent style should be included as inputs to our evolved ANN; Figure 5.15 shows the structure of our proposed network design. We have emphasized that Bayes' rule can be a powerful learning approach that can help us to analyse the past play information of an opponent, and determine useful information about each opponent's respective playing style.

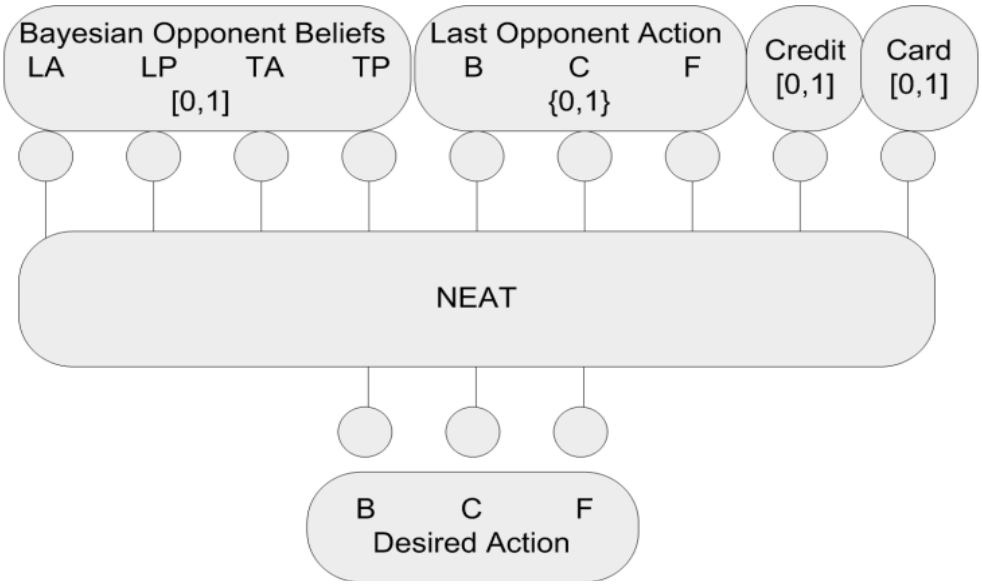


Figure 5.15: Design of proposed player

The usage of Bayesian probabilities to model uncertainties is popular in relation to imperfect information games such as Poker (Burns, 2006) and we feel that the information used to separate the analysis of each style of the opponent can prove useful to the evolution of our agent (Yao & Liu, 1998). The four per-style beliefs (LA, LP, TA, and TP) are used as four further inputs to augment the evolution of our player. Figure 5.16 displays the effect that opponent model augmentation has upon the evolution: The network quickly evolves to a solution which reaches a best tournament success of an impressive 97% over 100 tournaments. The same cannot be said of the average population performance, however. This stagnation of average best performance can most likely be attributed to the increased number of inputs, which increases the search space and increases the fragility of a good NEAT network to changes made by crossover and mutation. Although there were 400 generations of evolution, no further improvement was seen after the first 100.

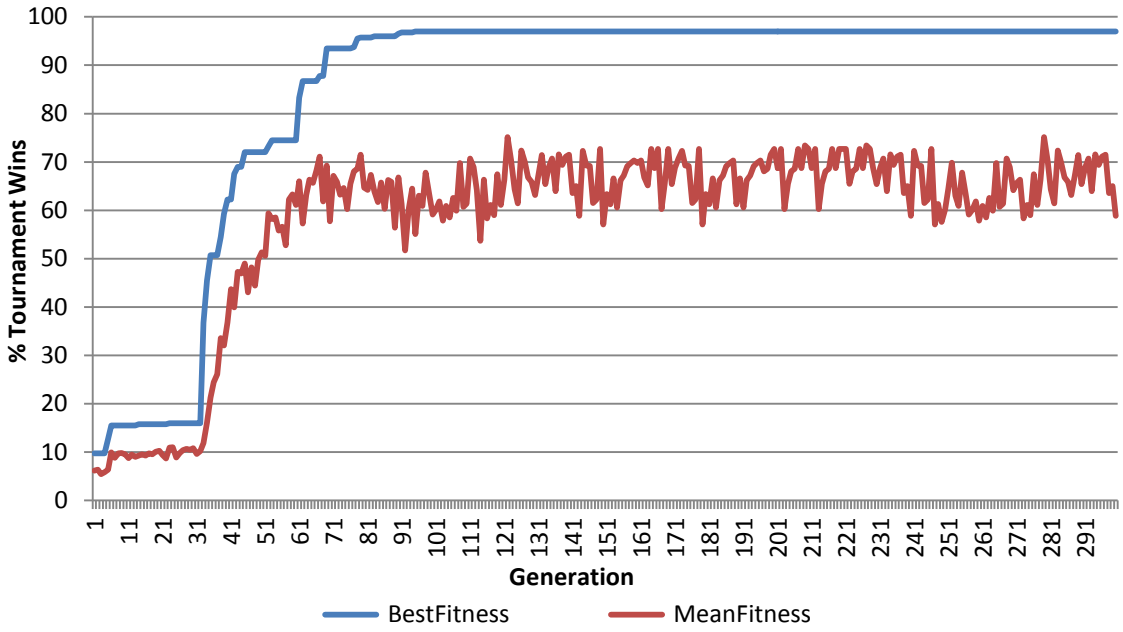


Figure 5.16: Evolution of a Bayesian Model-augmented ANN against all opponent styles.

Through analysis of our evolved neural networks, it was noted that the Bayesian probability inputs were weighted significantly in determining the output of the player. Contrasting the results in Figure 5.16 to those of Figure 5.14, the advantage gained through the use of Bayesian probabilities is significant; the initial stagnation associated with the difficulty of the evolutionary task shown in Figure 5.14 is expounded through the average of the best solutions not improving until well past the 120-generation mark, whereas using the Bayesian inputs opens up an 'evolutionary door' towards successful play within 35 generations.

5.3 Increasing the Number of Players

Much research into Poker has looked into simple two-player, one-on-one games (more commonly known as 'heads up' Poker) (Billings et al., 2003), (Koller & Pfeffer,

1995), (von Neumann & Morgenstern, 1944), (Barone & While, 2000), (Burns, 2006), (Billings et al., 2000) which reduces evaluation complexity, especially in relation to dealing with opponent models. The increased complexity of a 3 vs. 1 game, for example, calls for our neural network to interpret the information of 3 opponents instead of 1 (Sakaguchi & Sakai, 1992), (Saund, 2006). We believe that the ability of NEAT to evolve network topologies as well as weights might lend itself to such a problem, particularly due to the strong possibility of creating *recurrent* neural networks. Recurrence yields a 'memory' that can be utilized when data is fed sequentially (Elman, 1990), (Arvandi et al., 2008) (Gomez & Miikkulainen, 1998).

5.3.1 Recurrency Example

In order to show the benefits of the recurrent nature of the networks NEAT evolves, we evolve a network to determine the style of an opponent given a set of previous opponent actions. Our training data consists of four sets of data for each opponent style – within each style set are 100 sets of seven moves performed by a player of each opponent type. The aim of our network evolution is to give a set of seven moves to the ANN and have it determine whether the set is taken from an LA, LP, TA, or TP player. These sets are constructed from samples of each style of player whilst playing against one another. We evolve two networks – the first is given all seven action values in one instance over a collection of 14 inputs, every two representing a Boolean input determining if the action is bet [1, 0], check [1, 1], or fold [1, 0]. Four outputs are used in the network, one for each of the potential opponent styles; the output with the

largest output value is the style 'belief' of the network. This design is shown in Figure 5.17

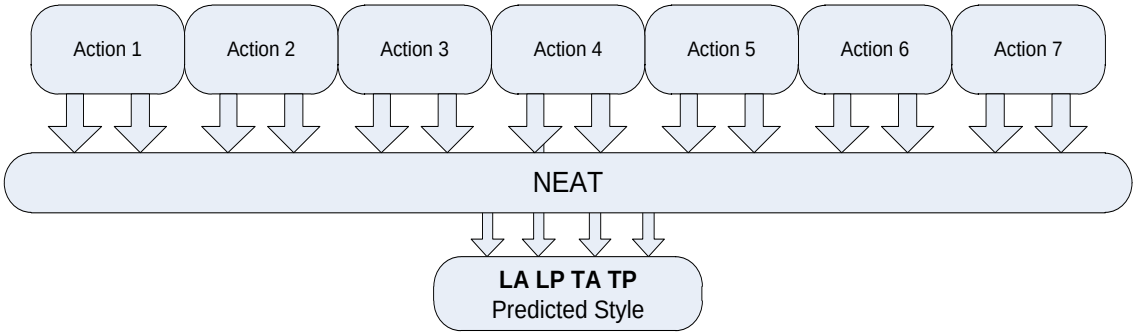


Figure 5.17: Design of a style-predictive ANN

Solutions are given a fitness determining the proportion of opponent styles it analysed correctly, which can be seen in Figure 5.18.

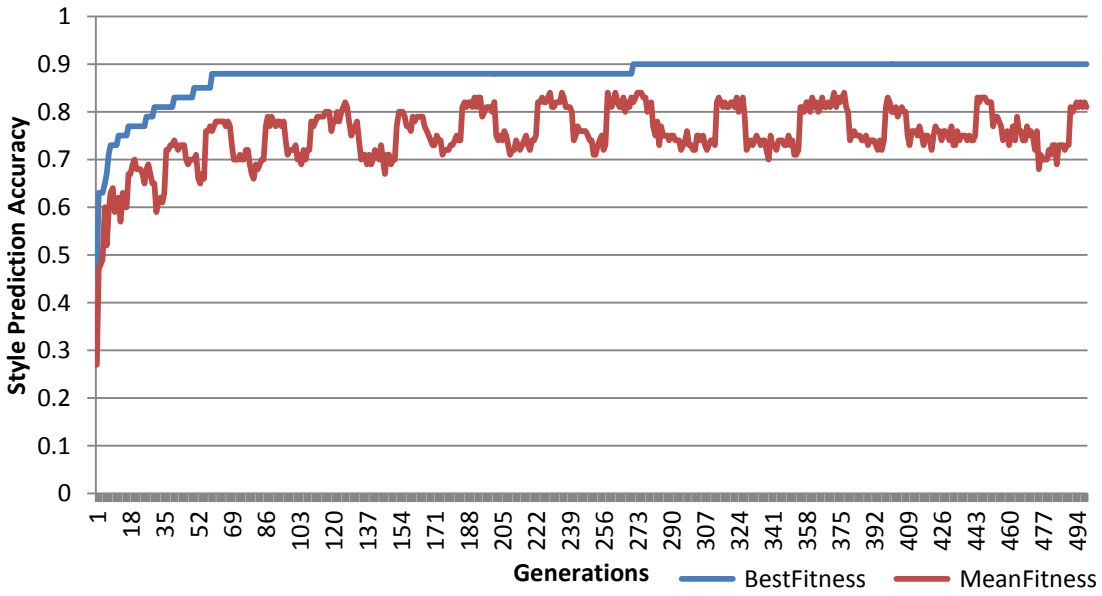


Figure 5.18: Evolution of a 14-input ANN analysing opponent style

The evolution displays a very high proficiency of accurate analysis (a maximum of 90% accuracy). The remaining 10% of inaccurate analysis can be attributed to the fact that the test data is gleaned from actual in-game performance of the given agent style, and as such the stochastic nature of play can sometimes yield a behaviour that would

belie an agent's natural style; for example, a Tight Passive player may hold a very strong hand, and will therefore be compelled to bet, a non-typical behaviour for a TP player. This non-typical behaviour can also be the contributing factor to the repeated pattern in the average population fitness in Figure 5.18; the evolution appears to be gradually progressing then re-evolving in a means to find an accurate distinction of actions sets which are of two separate types, but are indicative of a similar style of player. For the second network we evolve a solution which has *only two* inputs (which represent the aforementioned bet check and fold inputs), but pass each of the seven test data inputs *iteratively* to the network. In this scenario, we input the first action in the training set, then allow multiple steps of the network to transmit the data throughout the network. We then add the second, third, and subsequent n^{th} input. This staggering of the inputs between activating the network for multiple steps should potentially promote a recurrent memory in order to store (and subsequently 'remember') the previous actions and ultimately determine an opponent style (returned by the same four outputs as the previous experiment).

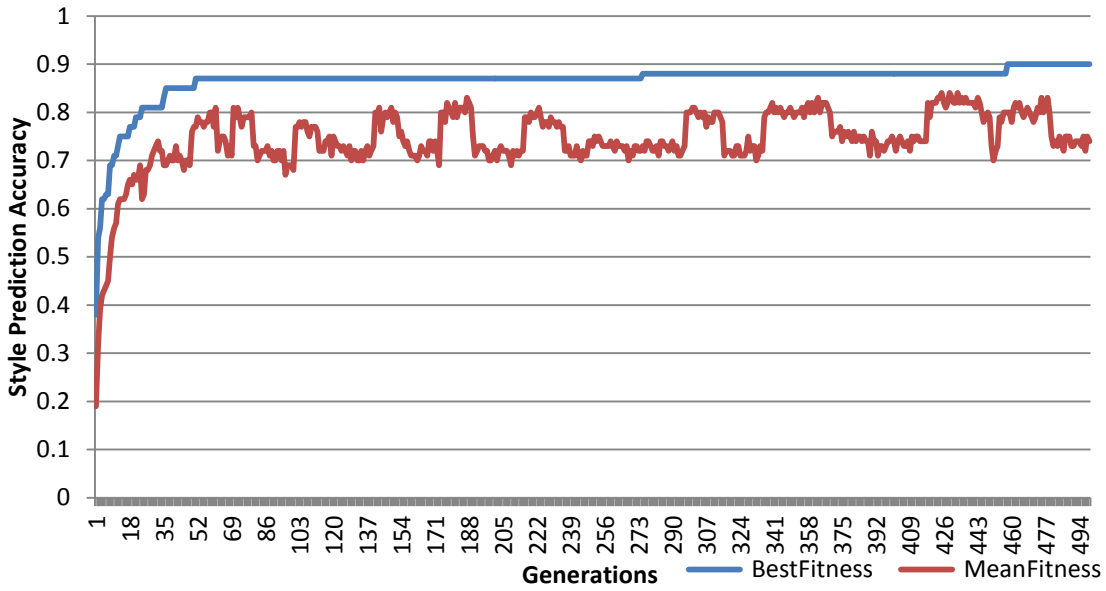


Figure 5.19: Evolution of a 2-input recurrent ANN analysing opponent style

As Figure 5.19 exemplifies, although the evolution takes almost 200 generations longer to reach the previous evolution's upper limit of 90% accuracy, the iterative nature of the inputs has yielded a recurrent ANN that determines opponent style accurately. Now that the effectiveness of a recurrent network has been demonstrated in a simple environment, our aim of reacting to more than one opponent shall take advantage of the 'memory' afforded to us through the recurrent connections. We use the same network inputs as Figure 5.15, and iteratively pass the inputs for the first second, third, and (potentially) *n*th opponent. Hence we can investigate whether this will result in a 'memory' of the previous opponents which should influence the current decision. After the final opponents' data is input, the output is received, and the action represented by the largest numerical output is performed. The fitness *f* for this experiment is represented by Equation (3), where *a* represents the number of player styles, *T* represents the fraction of tournaments won (over 100 tournaments), H_w represents the

number of hands won by the player, and H_p represents the total number of hands played.

$$\boldsymbol{f} = (\sum_{\boldsymbol{a}} \boldsymbol{T})\boldsymbol{a}^{-1} + \frac{\boldsymbol{H_w}}{\boldsymbol{H_p}} \quad (4)$$

The ratio of hand success is added in order to provide useful information to avoid large plateau in the fitness landscape due to the increased complexity of evolving a player against three opponents. Initial results from evolution without this value had difficulty in evolving initial reasonable strategies, in a fitness landscape that contained a greater number of large plateaus.

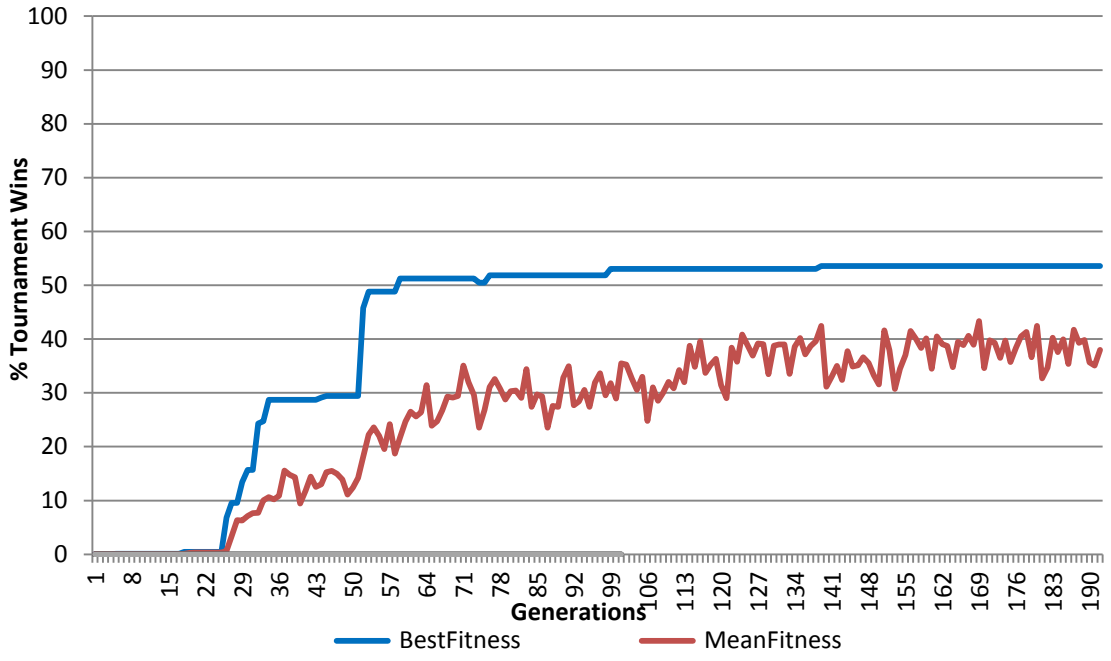


Figure 5.20: Evolution of a three-input recurrent network against all four types of opponent in a four-player scenario

As can be observed in Figure 5.20, the success of the player (without an opponent model) averaged over 5 individual SharpNEAT runs is modest, with a best tournament win percentage (over 100 tournaments) of 54%, and an average win percentage of

38%, modestly higher than the 25% which a set of 4 equally-matched players would obtain. Network structure in later generations fails to evolve to a suitable solution, arguably due to the lack of any further player-descriptive inputs. When we use Bayesian inference of opponent actions (Figure 5.21), however, we obtain an average success rate of over 70%, the same as that enjoyed by the same approach in the simpler 1 vs. 1 environment from fig. 10, and a best tournament success rate of 97%.

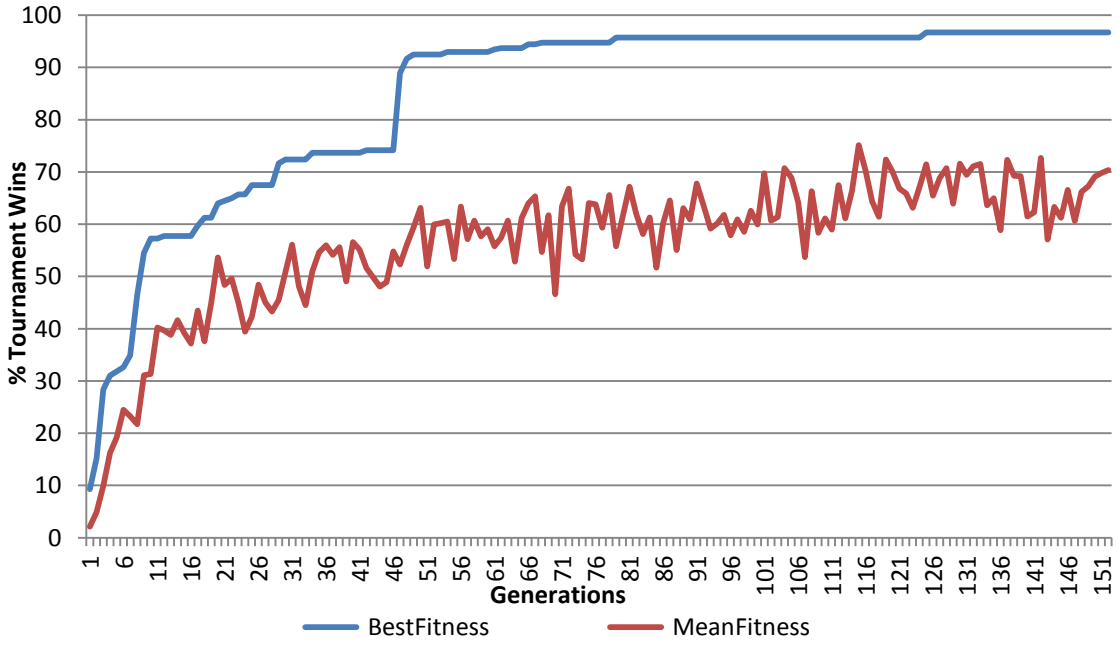


Figure 5.21: Evolution of a Bayesian-augmented ANN against all types of opponent in a four-player scenario

Looking at our network from Figure 5.22, we see few recurrent connections (although the bet output node is included once again), as well as a large improvement due to the opponent model inputs. One finding of note, however: In most of our networks which use the four Bayesian inputs, one of the inputs is almost always disconnected (in this case, input 6, the TP probability node). There appears to be no preference as to which one is disconnected, from which we can infer that the ANN

itself evolves to a simplified state through the fact that there are only three degrees of freedom for these four inputs, since their sum is always equal to 1 and as such the fourth input is somewhat redundant. The fitness of our evolved players was measured purely against tables of matching types; one table against three LA players, one against three LP players, one against three TA players, and one against three TP players.

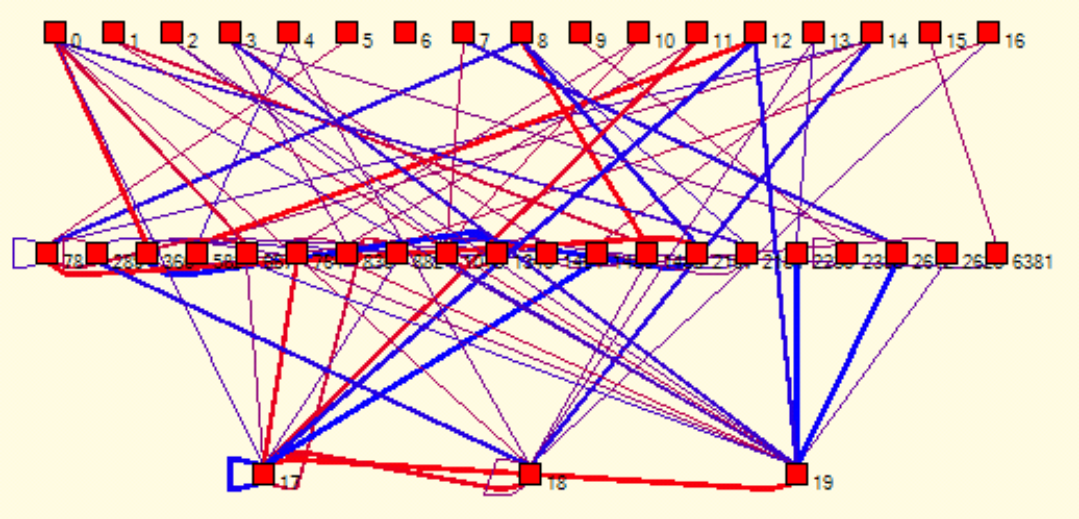


Figure 5.22: Network structure of an evolved network from Figure 5.21

In Figure 5.23 we illustrate the tournament success of the best solution from the evolution illustrated in Figure 5.21, against all possible mixed player-table combinations, and as we can see, the network has excellent success rates against all possible opponent combinations including those it *was not trained against*. This is arguably due to the advantage a player receives once it has access to beliefs about an opponent's style of play especially when we compare these results to Figure 5.20 where no opponent model is utilised. An issue of note with Figure 5.23 is the slightly lower success rates against tables of primarily loose players, including tables which consist of three LA, and three LP types, which it *was* evolved against. The nature of a 4-

player game of Poker differs from a 1 vs. 1 game, as a greater number of loose opponents mean that there is a greater probability of an opponent holding a good hand than if there was only one adversary. The choice of action in this case is therefore made much harder given the loose nature of all the opponents. Regardless, the win rates are still impressive against 3 opponents (when evenly matched players would expect to win only 25% of tournaments). Using Bayesian probabilities in this environment appears to be vital to the success of the player, especially when aiding the generalisation of our player against unseen combinations of opponent.

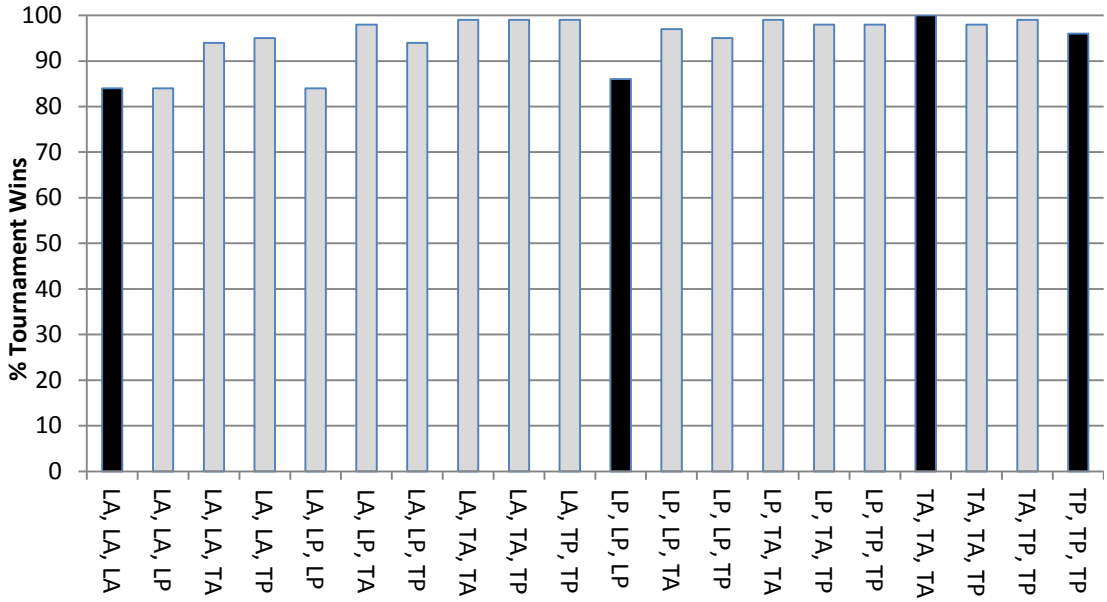


Figure 5.23: Tournament performance of the best evolved genome against all combinations of 3 opponents over 100 tournaments – The player was evolved against only the four highlighted combinations of players

The inclusion of an opponent model appears to aid both the evolution and decision process of the agent. This appears to be due to the ANN's ability to utilise the opponent model's separation of opponent types in order to determine a reasonable course of actions against the opponent(s). In order to test this theory, we re-evolve

after disabling different data inputs and evolved abilities available to our network in order to see how an evolved player can cope without such data. In this experiment we compare three inputs that we feel are essential to the players' function, namely opponent model information, last action information, and the recurrent nature of our evolved networks.

Figure 5.24 shows the difference in the fraction of tournaments won when each of the inputs of the player's network are disabled – this data represents the performance of the best performing network at the end of evolution. The greatest difference appears in relation to the removal of recurrence, and that of the most recent action by the opponent.

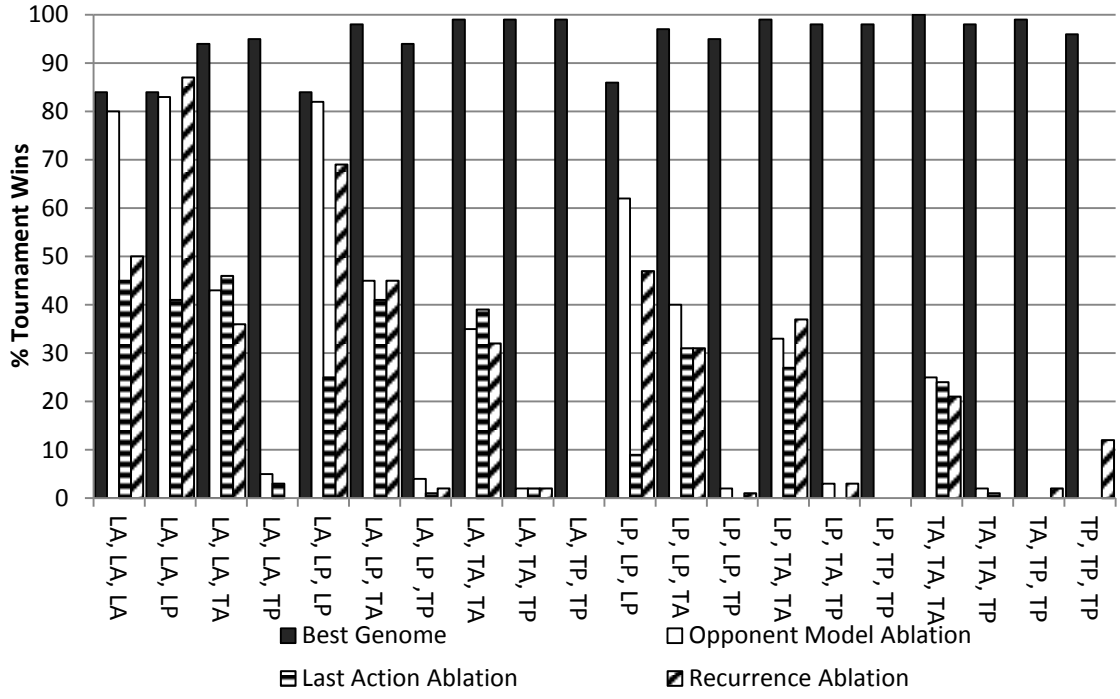


Figure 5.24: Tournament performance of ablated configurations of the 'best' evolved genome over 100 tournaments

The removal of recurrence removes the iteratively-passed opponent data, and hence memory of opponent characteristics, which greatly impairs performance. A noticeable facet of this is related to tables of three similarly-typed opponents; in Figure 5.24 we can see that in each of the situations where there are three opponents of the same type, the loss of recurrence causes a slightly less damaging effect. The removal of most recent action has a significantly damaging effect upon the success of our player, which is understandable due to the importance of taking our opponent's most recent action to infer the strength of their downcard. As for the removal of opponent model, the effect is again pronounced in all cases (although less so than the other two effects above). We believe that this means that the opponent model is integral to the operation of our neural network in terms of being able to determine a strategy tailored to the nature of the opponents. Our Bayesian description of opponents is a simple one, but the results are very clear; Bayesian probability-based opponent models can play an important part in creating (or evolving) a stronger Poker playing agent.

5.4 Dynamic opponents

In order to test the ability of our player to adjust to the potentially dynamic nature of opponents, we create two sets of tests: firstly we test our player (our best single ANN from the previous experiment, with all inputs enabled) against each of the standard four types of opponent, but we make our opponent bluff (bet when the player decides it should actually fold) with probability p, which is adjusted in increments of 0.01 where $0 \leq p \leq 1$. Figure 5.25 shows the results of these tests averaged over 100

tournaments per bluff probability. It should be noted that once the player has decided to bluff, the player will continue to bluff for the rest of the hand, normal behaviour is resumed (if deemed so by the bluff probability) in the next hand.

5.4.1 Bluffing

"Bluffing" improves the performance of a TP player: at a bluff probability of 0.12 our best evolved player drops to a (still impressive) 72% tournament success rate. This is understandable, as there is strong evidence that tight approaches are greatly strengthened by a level of bluffing (Koller & Pfeffer, 1995), (von Neumann & Morgenstern, 1944). Indeed, Koller and Pfeffer showed that Nash equilibrium players bluff.

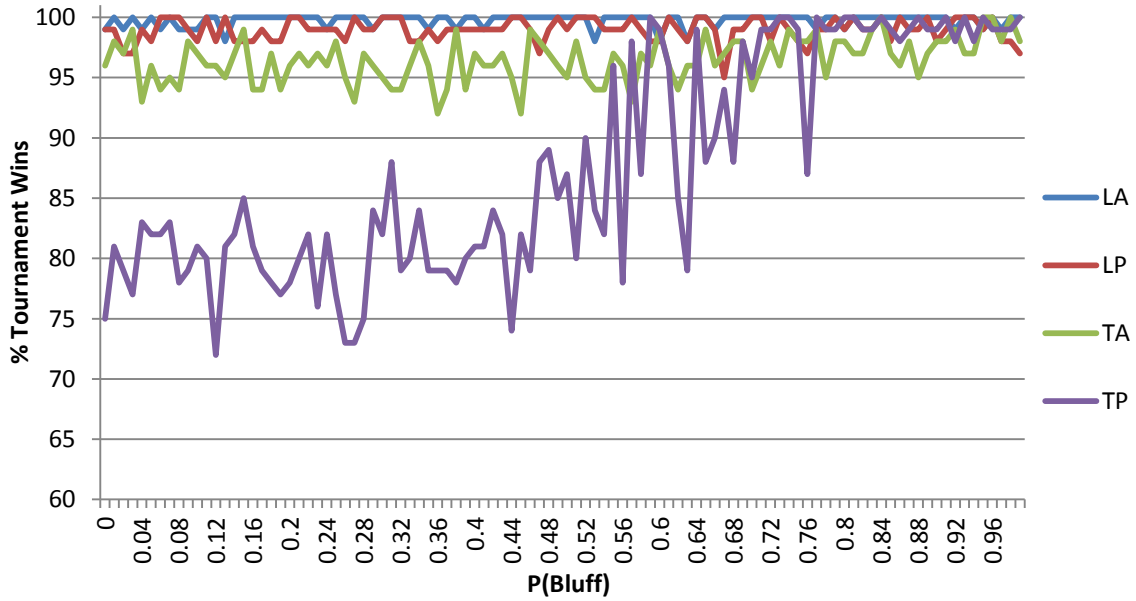


Figure 5.25: Tournament performance of the 'best' evolved genome against partially randomized opponents over 100 tournaments

As the chance of performing a bluffing action rises, and the TP player becomes "looser", so does the success of our player against the bluffing TP player. In this case,

our opponent model now assumes that the opponent is Loose Aggressive, and caters for the eventuality that our randomized player will be bluffing a large portion of the time. As for the other types of opponent, they will all mostly be classified as being an LA player as *p* rises. Figure 5.26 represents the probabilistic belief of opponent style of our Bayesian network against a TP player given varying levels of bluff probability – the probabilities represented are the belief after one single game against the opponent. We can see that there is a gradual trend towards the belief that a TP player is LA as the probability of bluffing increases (with some exceptions, but the random nature of the actions can lead to some unexpected analyses, as well as the stochastic nature of the player cards themselves). The most important aspect of this analysis is that even though the Bayesian analysis is not necessarily accurate 100% of the time, the play strength of our evolved agent is still strong enough to be able to deal with the misinformation. Indeed it is true that given a run of similar (high) cards it is hard to distinguish between loose and tight players. Bluffing does not improve play quality of LA, LP or TA players, as we observed in Figure 5.25.

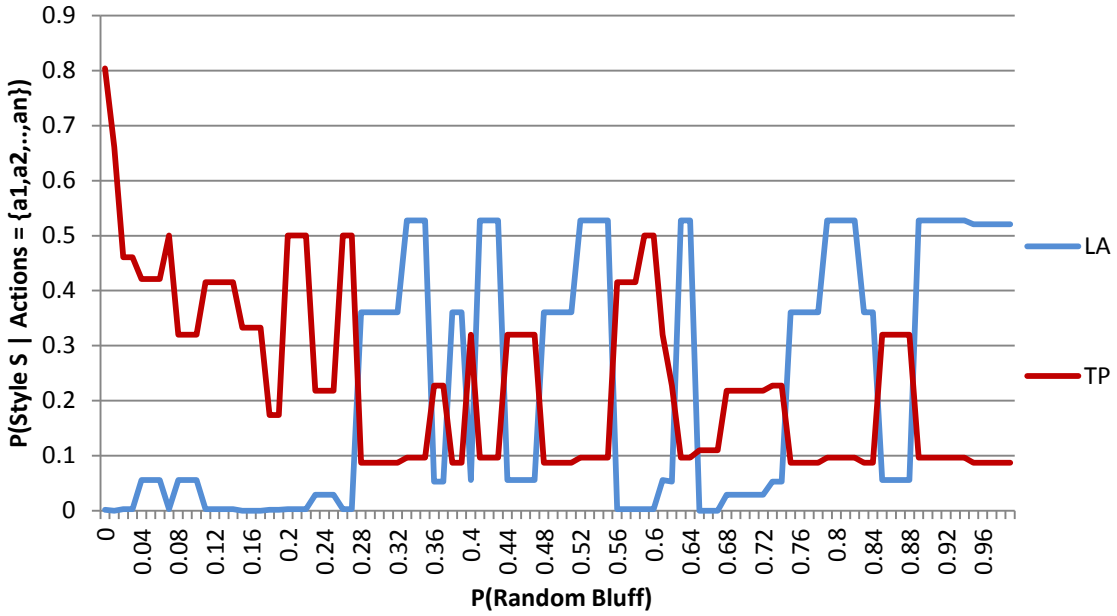


Figure 5.26: Bayesian analysis of a TP player with varying bluff probabilities

5.4.2 Play Style Adaptation

If a players' current tactic is unsuccessful, then it is sensible for the player to change their current strategy; we implement a series of players that transition from one style to another once their chips are at a level that is lower than 50% of their initial chips at the start of the game. We can see in Figure 5.27 that our player is able to cope well against our style-changing opponents. It is notable that our player is most susceptible to opponent strategy change when moving from a looser style to a tighter one. The main reasoning behind this is that our Bayesian modeler has to readjust its style-representative weights in order to accommodate the opponent's shift in style, and as such a lag is involved in "understanding" the opponents' strategy.

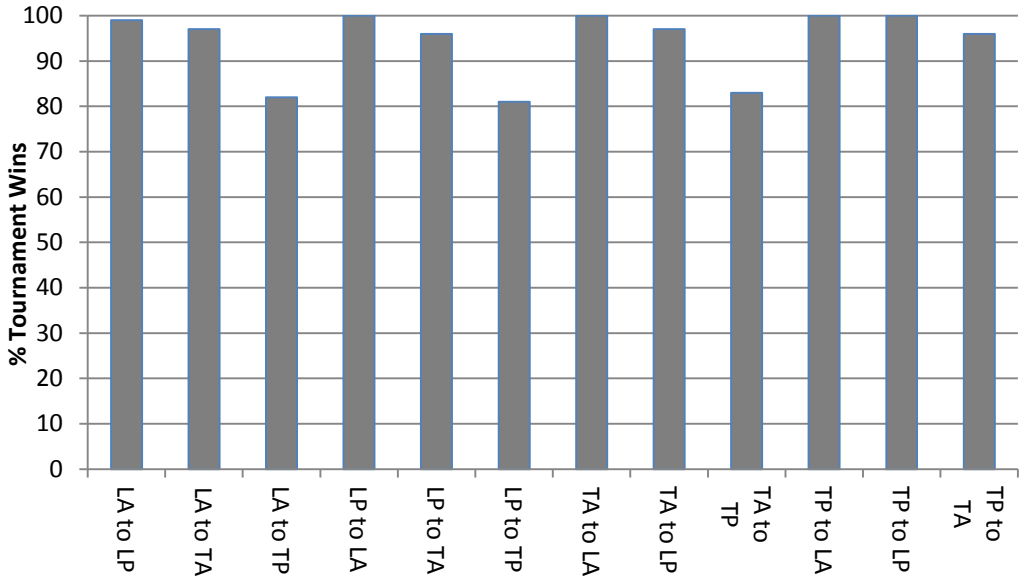


Figure 5.27: Tournament performance of the 'best' evolved genome against dynamically styled opponents over 100 tournaments

Moving from a tight style to a loose one can be a good way of scaring opponents out of the game until they realize the change in strategy. This approach is important in human play of Poker, in order to avoid predictable play, as well as exploiting the gullibility of the opponent (Sklansky, 1992). The failing of this approach against our ANN strategy, however is that the probabilistic way in which our modeling approach updates its beliefs means that these beliefs will be altered significantly when a tight player repeatedly performs an action that it should rarely do (the main example being to move from TP to LA, drastically increasing the frequency of betting actions; see Figure 5.23).

5.5 Discussion

The collection and creation of action probability data is a straightforward task in a variety of games. Using a hard-coded or parameterized set of possible opponent

strategies and Bayesian analysis we determine opponent style based upon previously observed actions. The results from Figure 5.23, Figure 5.25 and Figure 5.26 show that this approach works even when the opponent strategy is not in the set of possible opponent strategies. Bayes' theorem, which we have shown effective for Poker, has the potential for broad application of this approach in other games (including video games) where discrete actions can be related to strategies, or short term goals.

Bayes' theorem can be utilized in conjunction with approaches other than neuroevolution; for example, if we have a Finite State Machine (or any hard-coded) agent, a switch between states can be prompted by Bayesian beliefs to select suitable counter strategies (as has been utilized in (Baker & Cowling, 2007)).

The strategy selection in this approach is generally smooth in relation to the gradual updates to the probabilities (see Algorithm 4.1), yielding a sensible change in strategies as the game situation changes (as long as the hard-coded strategies are sensible and broad), as well as being relevant to the player's actions, which may give the AI opponent's play a more self-consistent, natural feel.

This approach assumes that we have a collection of known strategies to compare against, but as shown in this paper, our models need only represent a spectrum of possible opponent models (which need not include the actual player strategy) that do not necessarily have to be entirely correct or complete to be useful. It is possible that these strategies could be obtained directly by observation of actions of human players and their effective (human) opponents.

5.6 Summary

In this chapter, players are evolved using various approaches for a simplified game of Poker. We first show it is easy to evolve a player against individual opponents of a fixed style. We then investigate the utility of opponent models in aiding the evolution and performance of game-playing agents against players that do not make use of such information.

Against a single adversary, the results show little difference between approaches that use an opponent model, and those which do not. However, the benefits of using opponent models are much greater when facing increased numbers of opponents. Bayesian opponent modeling is shown to be a crucial input to a neuroevolutionary player, as is the recurrence which allows the evolved networks to have short term memory. Furthermore, the approach is able to generalise and defeat tables of opponent combinations not yet encountered.

We test the approach against opponents that employ simple bluffing tactics, as well as simple dynamic strategies. In these experiments we find that our opponent-model augmented NEAT networks are able to perform well against these dynamic opponents, which do not lie in the set of possible opponent strategies for the opponent model.

Chapter 6: Adaptation to Nonspecific Opponent Styles

In this chapter, we take the strongest evolved Bayes' rule-utilising ANN from the previous chapter (evolved during the series of evolutions in Figure 5.21) with the intention of gauging the response of the Bayesian player against a series of agents that are designed to not fit in with the typical definitions of 'Loose Aggressive', 'Loose Passive', and so on. We evolve a population of GA players against the Bayesian player, with a greater complexity than that of the opponents the ANN was evolved to face, and subsequently Coevolve two populations of opponents to attain the strength of the model in attaining a suitable response to the perceived style of a non-style specific agent.

6.1 Evolutionary Opponents

In order to test the robustness of our best evolved ANN, we devise a player which based upon an evolutionary algorithm with a more standard (non-neural network) representation. In this sense, 'robustness' describes how well our opponent model and ANN copes with opponents that do not fit within the aforementioned styles (LA, LP, TA and TP). We evolve a set of weights which give a similar representation to that of Figure 4.1, but with finer granularity. We use a chromosome that consists of three sets of α and β values to represent each card, each one of the three sets representing the opponent's previous action; one set for a previous bet, one set for a previous check,

and one representing a new hand. This chromosome is identical to the one described in Figure 5.1. For each card, α is used as the probability of staying in the hand given the card strength, and β is used as the probability of making a betting/raising action given that the player stays in the hand. The action is chosen based upon the α / β values and the card held (as shown in Figure 4.1). In our evolution we use a population size of 100, a mutation probability per allele of 0.01, and single point crossover. These values are chosen such that with 60 alleles per chromosome, on average one mutation per chromosome per generation is made. Mutated alleles are normalised to keep all values between 0 and 1. Figure 6.1 shows the evolution of this description against our neuroevolutionary player.

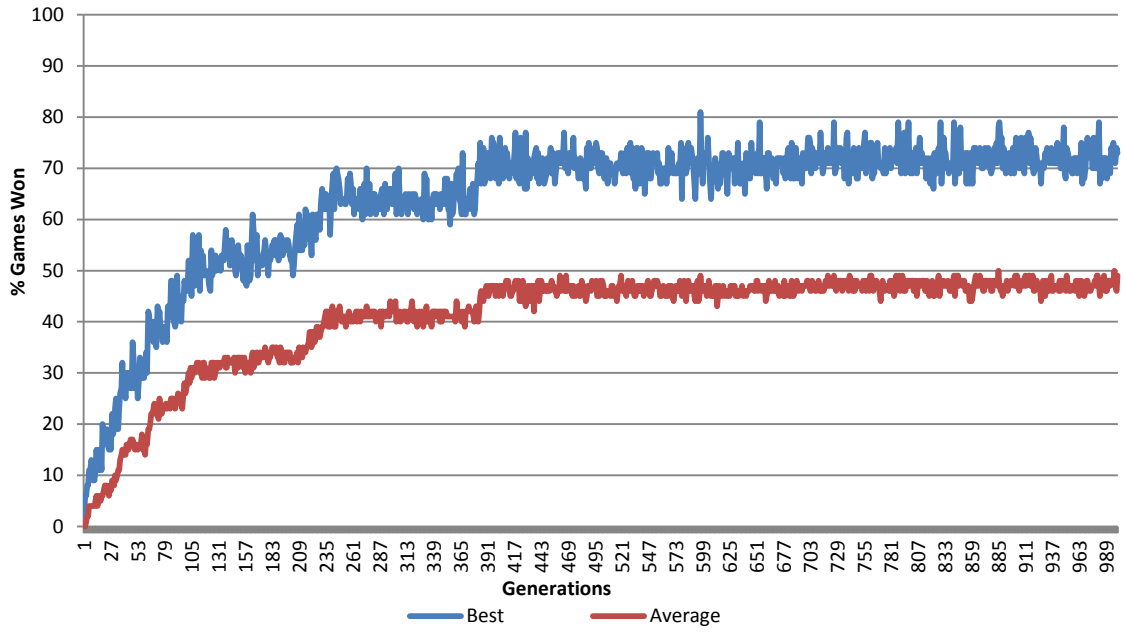


Figure 6.1: Evolution of a probability-controlled GA player against the best evolved ANN

As we can see, our evolutionary approach yielded a player (the GA-Player) that can (at best) beat our ANN-evolved approach 81% of the time in heads-up play. Further

evaluation over 10, 000 games shows that the best genome actually has an average success of 63% wins against our Bayesian ANN. The GA requires over 100 generations of evolution to reach a point at which it can beat the ANN in 50% of tournaments played. This can be attributed to the ANNs response to its opponent model adapting to the changing nature of the GA-Player. Looking at the evolved α and β values, we have observed that the GA evolves to play within a mid-ground between an extremely tight game, and a light bluffing approach – Figure 5.25 suggests that light bluffing with tight play can be effective against our player; the representation of our GA means that a bluff probability can be set for every strength of card for each action our player can perform. This gives each chromosome the evolutionary potential to yield a strong player. The Bayesian inputs to the ANN in relation to the evolution of the GA exemplifies this to the extent that the belief of opponent style represents a Tight Passive opponent, therefore meaning that the minor amount of bluffing afforded by the probabilistic GA representation means that a slight advantage can be taken through the external representation of holding a strong card and playing tightly, although a relatively weak card will be held. In response to this, we once again evolve the Bayesian ANN against all four original styles, but also against the best evolved GA solution. The results of this experiment can be observed in Figure 6.2.

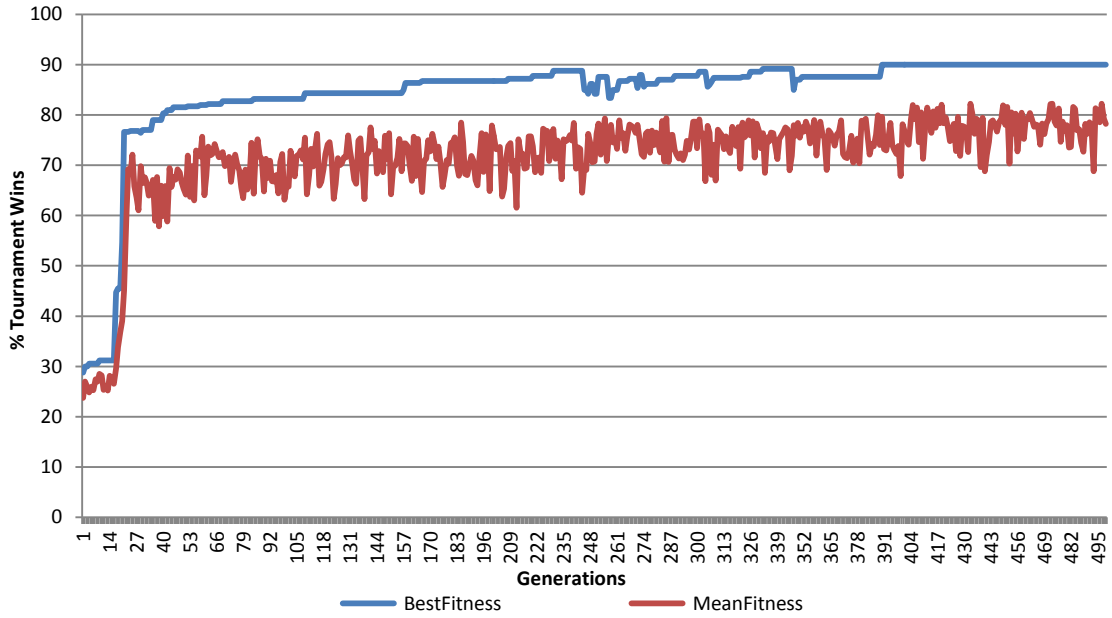


Figure 6.2: Evolution of a Bayes' rule Utilising ANN against the GA player

We observe that the ANN has evolved very strongly against the GA player (in addition to the original styles) with a greater average fitness than in Figure 5.21 – In a separate experiment, evolving the ANN against the GA on its own yields a >70% win rate within 20 generations, suggesting that the GA approach was overfitted to our original ANN approach.

6.2 Coevolutionary Challenges

Section 5.3.1 displays that the evolved Bayesian ANN is robust in its performance against a *collection* of opponents that have not been evolved against. Our approach has yielded the conclusion that even if an opponent is not strictly within the definition of an opponent *style*, as long as the model can generate a belief that the opponent's behaviour is at least within the parameters of one of these styles, performance will be

improved over an agent employing no model. To test this belief, we use a Co-Evolutionary process to evolve two randomly-initialised populations in order to generate a collection of potentially effective, yet not necessarily specifically styled, agents.

6.3 Chromosome Representation

The chromosome design is influenced by two main factors; these are the opponent's *previous action*, as well as the strength of the *card held*. For each possible card held, we have two values; α and β, which have definitions synonymous with the values used in the construction of our simple player from Chapter 4, Figure 4.1. Alpha (α) represents the probability of staying in the hand given the current card (Fold/Stay in Hand), and Beta (β) represents the probability of making a betting action given that the previous probability has determined that the agent stays in the hand (Bet/Check), using the design from Figure 5.1. There are three α and β values for each card, each one representing the chromosome's response to the previous action of an opponent; one pair in response to a betting action, one in response to a checking action, and the final pair is in response to a new hand (this, in theory could be attributed to an opponent not having yet performed an action in this hand). The sequence of actions in determining the agents response is as follows (with respect to the allele position *P* in the chromosome):

- Check the opponents previous action (*Bet*: *P* = 0-19, *Check*: *P* = 20-39, and New Hand: *P* = 40-59)

- Check the α and β values for the card held (assuming *Bet* was chosen previously) (Card 0: *P* = 0+1, Card 1: *P* = 2+3... Card 9: *P* =18+19)
- Generate a random number in the range 0.0 to 1.0 , *r*
- Assuming Card 9 held, if *r* < *P* where *P* = 18 (α), then the agent stays in the hand (else fold). If *r* < *P* where *P* = 19 then be bet (else check)

6.4 Coevolution

The approach used in this section is inspired by the Evolutionary technique introduced by Rosin and Belew's investigation into competitive CoEvolution, concerning itself improving agents for the game environments of Tic-Tac-Toe and Nim (Rosin & Belew, 1997). Two populations of equal size are generated, within which each solution is measured in fitness against each member of the *opposite* population. Candidate fitness is evaluated using the concept of *shared fitness* which is such that weaker solutions with important, individual, attributes have a fair chance of survival (regardless of certain weaknesses). The fitness of a solution *s* within Population *A* is determined relating to the number of solutions *s* has defeated from population *B*, with respect to how many other solutions from Population *A* have also defeated that player (i.e. if a generally weak player is able to beat a single opponent that no other player can defeat, it has therefore a fairer chance of surviving due to its individuality). Figure 6.3 (overleaf) gives an example of how this fitness is implemented.

```
// For each member of Population A
for (int i = 0; i < POPULATIONASIZE; i++)
{
   // Initialise fitness
   fitness[i] = 0;
   // For all opponents solution i has beat from population B
   foreach (Opponent j in OpponentsIBeat)
   {
      // Fitness is incremented relating to the number of members
      // of Population A that have also defeated Opponent j
      fitness[i] += 1 / 1 + MembersDeafting[j];
   }
}
```

Figure 6.3: Pseudocode of the Shared Fitness Function

The algorithm mainly takes into account *individual* successes to aid potentially successful solutions. We further implement a *Hall of Fame* system, which keeps the best solution from every previous generation, and is included in the number of solutions candidates have to face each generation (i.e. after evaluation against all members of population B, the collection of Hall of Fame members must be evaluated against also). The number of members of the Hall of Fame defeated by other members of the same population is also taken into account when evaluating fitness. This is such that a cyclical evolution is prevented where Type B beats Type A, Type B is defeated by Type C, but Type C can be defeated by Type A, which could then subsequently be re-evolved. A Hall of Fame avoids this problem, because all three types remain in the population. Obviously, performance issues are incurred by such a practice, as further generations occur, the greater the reservoir of old players to be tested against.

In order to reduce the performance impact of the greater number of Hall of Fame members over further generations, it is determined that a random sampling of the Hall of Famers for evaluation will be taken each generation (a selection equivalent to half

the population size), although there is difficulty in assuming the best trade-off between performance and accuracy.

The Coevolutionary process is constrained by the fact that a 'winner' and 'loser' (i.e. solution 1 *beats* solution 2) is required. In this case, playing 100 games of One-Card Poker between two solutions can render numerous variations of results, as well as presenting the problem of which percentage of wins constitutes a definite *win* over an opponent. The difficulty is compounded when considering the situation that 55 percent of wins in the favour of player *A* could represent a particularly unlucky spell of games for the other player *B* (in which case *B* is as good as *A*), or conversely a unlucky spell for *A* which is actually much stronger than *B.* We implement a *threshold* value *T*, which determines that a solution *a* has defeated solution *b*; if solution *a* has won by *T* games or greater, then *a* has defeated *b*. Any win margin falling below the threshold for both sides indicates a 'Draw', and dictates that a judgment of 'both win, both lose' is taken i.e. there is both a loss and a gain for tying with another solution, which should render the advantages to either side moot in determining fitness. For the given experiments, the threshold has been set to a size of $T = 10$. The subsequent experiments are run on a quad-core Intel Xeon X5460 running at 3.16 GHz with 4GB RAM using C#.NET 3.5 running under Windows Server 2003 R2. We Co-Evolve two populations, each of size 100, for 550 generations using the fitness described in Figure 6.3; traditional single point crossover is used, with a mutation probability of 0.02 per allele of the gene. This is such that approximately one mutation is made per chromosome per generation. Each population is sorted using fitness-proportionate

(Roulette wheel) selection, and the top half of solutions is kept, whilst the second half is populated with the mutated and bred solutions from the remaining top half. The results of this Co-evolution can be seen in Figure 6.4.

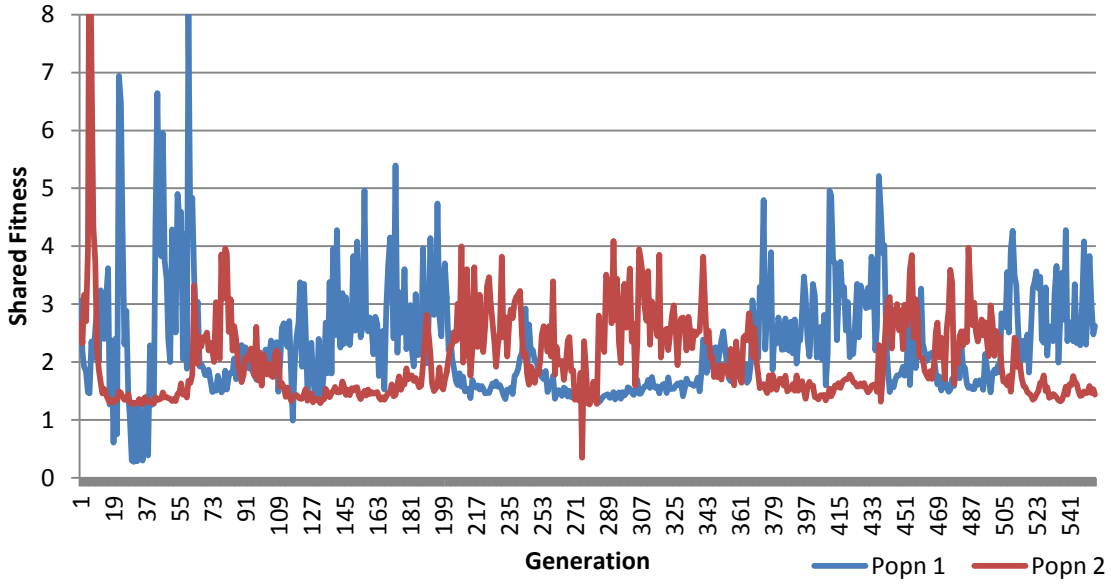


Figure 6.4: Coevolution of two populations using a 'threshold' size of 10

As is generally evidenced by coevolutionary approaches, the cyclical process of a gradual arms race develops. The nature of the shared fitness is such that rather than seeing an overall gradient of improvement in the best results of each generation, the fitness values reached are within a small margin (from a value of 0 to 8, as we can see here). It can be seen that the margin between the performance of one population is not significantly correlated with the performance of the second (such that when the fitness of Population 1 improves, that of Population 2 degrades), this is understandable in that the populations are also evaluated against the Hall of Fame members, which would further skew the resulting fitness of the population. The wildly erratic fitness values at the start of evolution (where Population 2 reaches an extremely high fitness)

can be accredited to the general weakness of the randomly initialised solutions; once a population-relative successful solution has been found, its performance (when contrasted with weaker members of the population) would have a significantly greater chance of defeating solutions others in the same population could not, thus yielding the excessively great fitness. The overall competition between the two populations appears balanced, with neither population taking an excessive advantage over the other, nor either population remaining dominant for a long period of time.

As a measure of the behaviour and performance of each of the evolved best solutions from each generation, we play each solution against the original static styled opponents, as defined in Chapter 4, Section 4.2.1. The results of this evaluation can be observed in Figure 6.5 through Figure 6.8).

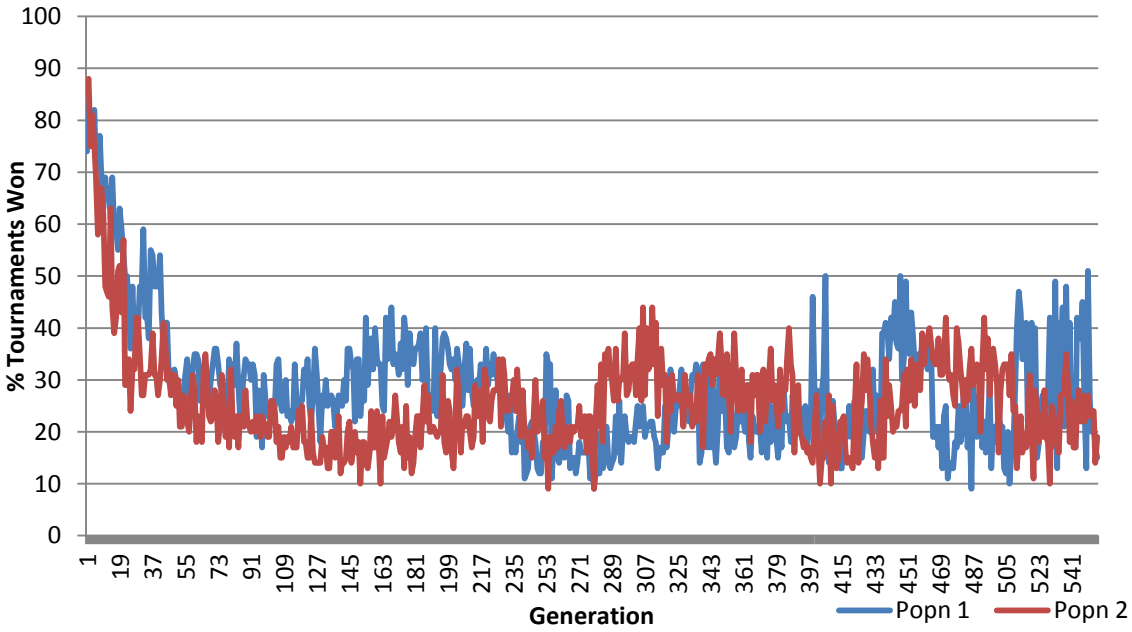


Figure 6.5: Performance of a static LA player against the Best evolved solutions from each generation.

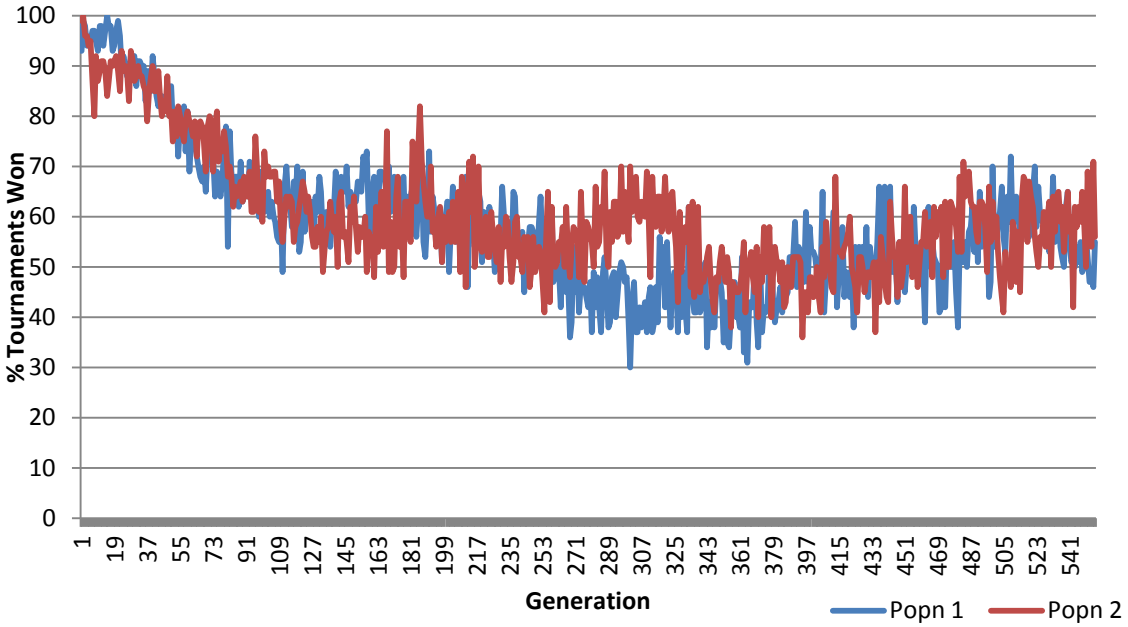


Figure 6.6: Performance of a static LP player against the Best evolved solutions from each generation.

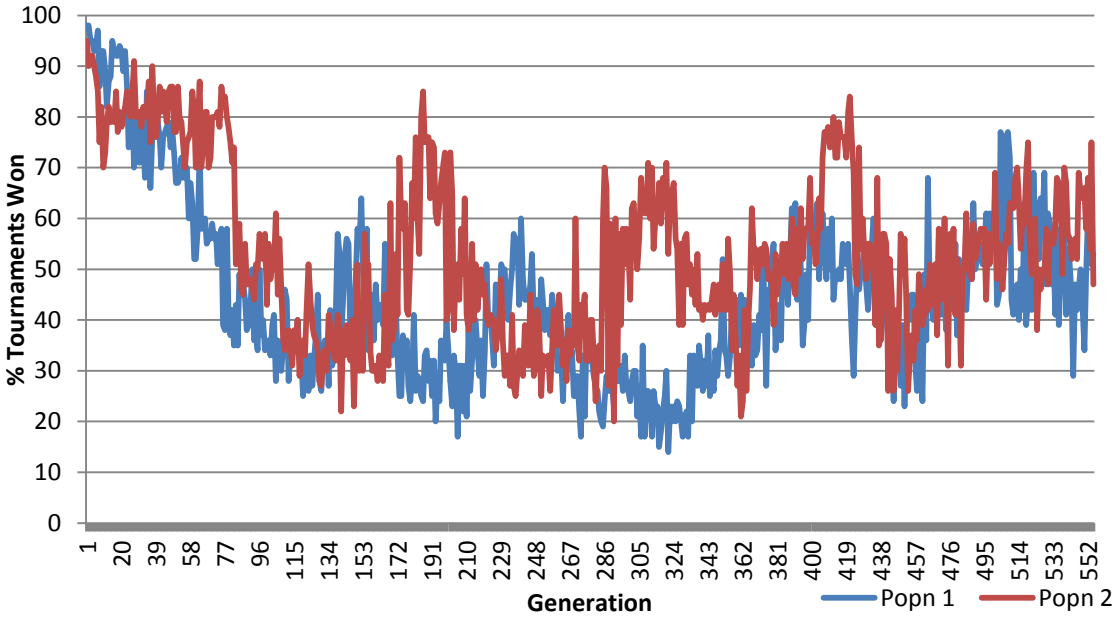


Figure 6.7: Performance of a static TA player against the Best evolved solutions from each generation.

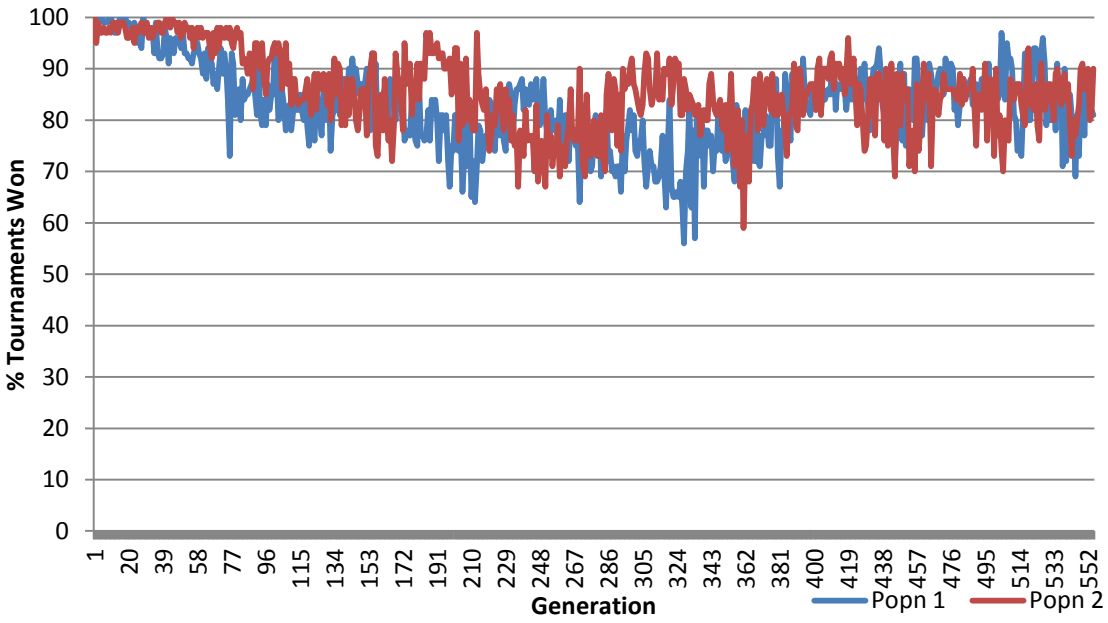


Figure 6.8: Performance of a static TP player against the Best evolved solutions from each generation.

As we can observe, the result is somewhat typical given the representative 'strength' of each of the static styles (discussed in Section 4.3.2); each 'best' player performs poorly against each of the styles, and performance gradually improves as the populations coevolve. It should be noted that the solutions are not evolved against any of these styles, only against one another. Most noticeably, there is very little improvement against the Tight Passive player in Figure 6.8 as the solutions evolve against one another. This is arguably due to all the players evolved against each other are typically of a tight variety; the aggressive players will be beaten easily through tight play, but when playing against a passive player, it is difficult to measure strength of cards due to the lack of betting, this also attempts to explain the erratic performance of the solutions against the LP players.

Given our ‘tightness’ hypothesis (i.e. the evolved populations of co-evolved players are predominantly tight players), we shall measure this against our evolved Bayes-ANN player to determine both the resilience of our model, and the style predicted by the Bayesian analysis against both populations of evolved players. Figure 6.9 and Figure 6.10 display the performance of the Bayesian ANN, as well as belief of the opponent style of each of the evolved opponents from each generation over a series of 100 tournaments per solution. These graphs are ranked by play style (as decided by the Opponent model) as well as by performance against the Bayesian ANN.

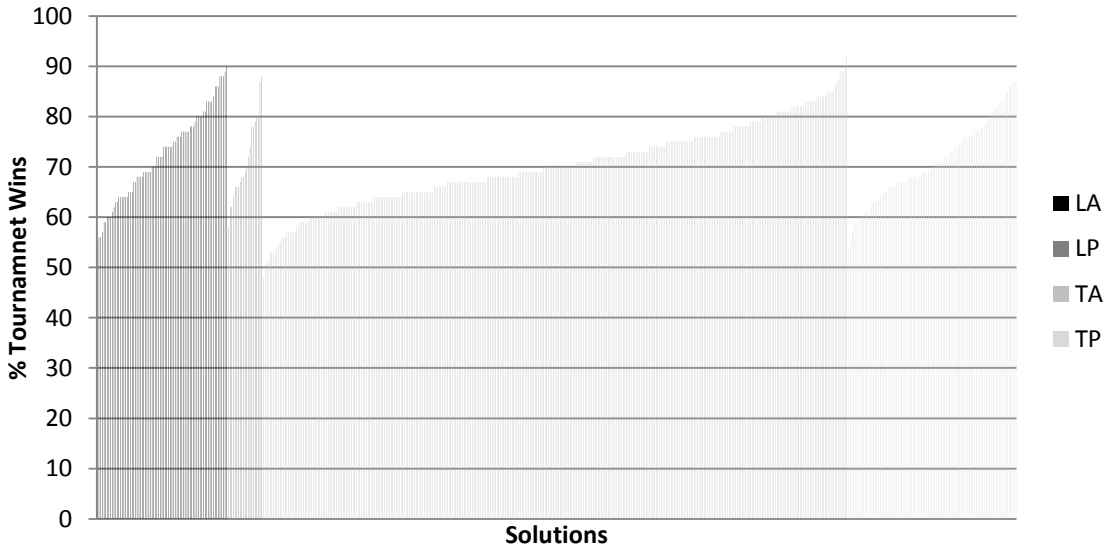


Figure 6.9: Performance of the Bayesian ANN Player against the Best evolved solutions from population 1.

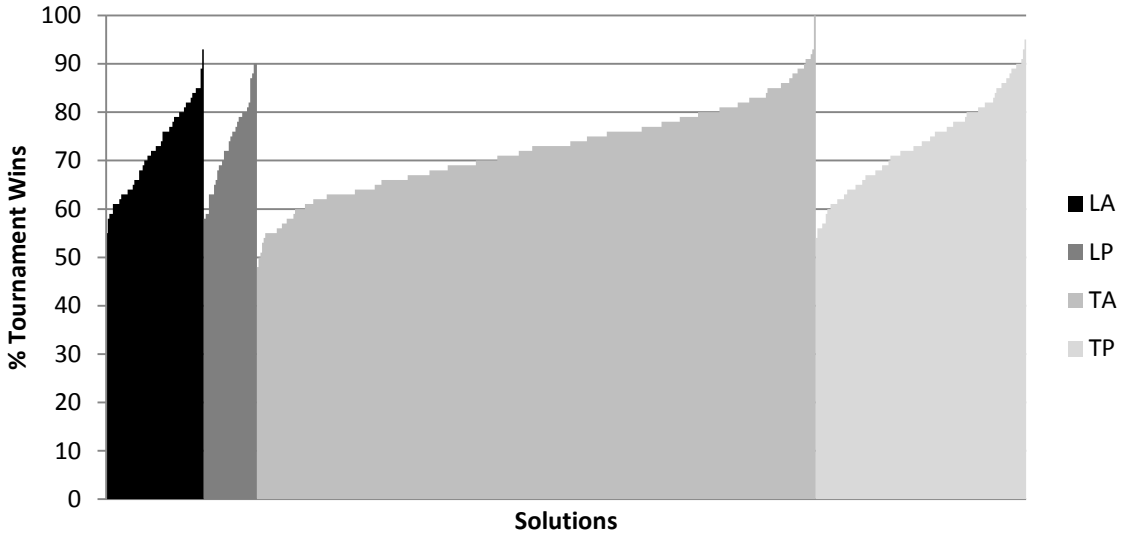


Figure 6.10: Performance of the Bayesian ANN Player against the Best evolved solutions from population 2.

From both Figure 6.9 and Figure 6.10, the assumption that most of the evolved solutions are tight styles of player has rung true. The majority of evolved solutions in both populations are judged to be within the Tight Aggressive label. Furthermore, only two solutions from each population were able to reach a success rate of over 50% against the Bayesian ANN, all of which are assumed to be within the Tight Aggressive style bracket. This evidence is further supported by the results from Figure 6.1, as the best evolved players employed a tight approach to play, with a loose bluffing approach when holding weaker cards. Whereas the opponent model assumed a Tight Passive style of player for the solutions in Figure 6.1, a Tight Aggressive style is used by the best solutions here. Once again, through evaluation of the best performing solutions, we see a tight approach to play, but a slightly greater probability towards bluffing than that seen in Figure 6.1's solution. This is a reasonable explanation why the model believes the solution to have a Tight Aggressive style, due to the frequency of bluffing being slightly greater.

Irrespective of the trend towards Tight Aggressive players being produced by the coevolution, a large selection of varied players has been produced in order to test the robustness of the previously evolved Bayesian ANN, of which the ANN performed competently against all but four solutions. The weakness seen in the evolved solutions against a static Tight Passive opponent in Figure 6.8 has proved to be a limiting factor as to the performance of the evolved solutions, but an initial 'tutoring' stage could be applied to any further investigation into the evolution of testing candidates. A tutoring stage would potentially include the standard 'LA, LP, TA and TP' styles within the Hall of Fame set, and subsequently promote evolved solutions to play strongly against as many opponent types as the chromosome would allow (This is further discussed previously in Section 5.1.1).

6.5 Summary

We implement a GA-controlled player which uses a concise, play-probability oriented structure which gives good control over the probability of bluffing. This player representation is evolved against the best evolved Bayes' rule-utilising ANN from Chapter 4 and performs strongly given a good deal of time for evolution; when re-evolving our Bayesian ANN however, the ANN is able to quickly adapt to the GA player's play, as well as the original four styles of opponent, providing further evidence that opponent models remain useful even when the opponent strategies lie outside the set of modelled strategies.

A Coevolutionary approach is presented with the aim of evolving a pair of populations which represent a collection of strategies against which the Bayesian ANN is not familiar. The populations of solutions prove to play adequately against the set of four static play styles defined in Section 4.2.1. We observe the performance of the Bayesian ANN with a particular reference to the play style perceived by the opponent modeling technique as to the style of the solution. Through the coevolution we find that a diverse set of opponent styles is evolved, as evidenced by the beliefs of our opponent model. The performance of the Bayesian ANN proves to be strong against a vast majority of all evolved solutions, once again exemplifying the strength of an opponent model in inferring a belief of an opponent's strategy, even when the opponent itself is not strictly defined within the parameters of a defined 'type'.

Chapter 7: Texas Hold'em

This chapter investigates the expansion of the work from Chapters 4 and 5 from the single-card game to full-scale Texas Hold'em Poker. We will once again analyse the actions of an opponent in order to use the resulting probabilistic model to aid the decision-making capabilities of a Poker-playing agent. The chapter is structured as follows: We present the game of Texas Hold'em Poker and describe our experimental conditions for evolving an ANN-controlled agent. We then apply our opponent model to the evolution, and cover the significant changes required to cope in a multi-round betting environment. Furthermore, we test the resilience of our evolved agents in a multi-player scenario as well as against dynamically performing agents.

7.1 Texas Hold'em Poker

Texas Hold'em Poker is arguably the most popular form of Poker played around the world today overtaking other forms of Poker as the most popular in casinos worldwide (Clark, 2006). The popularity boost of Texas Hold'em is such that the game has permeated into other forms of popular culture; In the Ian Fleming novel 'Casino Royale' for example (Fleming, 1953), British agent James Bond plays *Baccarat* against the primary antagonist, Le Chiffre. In the most recent adaptation of the novel into film, however, the game has been changed to Texas Hold'em, primarily because of the worldwide popularity the game has found (MGM, 2006). The three main actions performed in-game are common to all forms of Poker, as follows:

- ***Bet/Raise***: Add money to the pot, and increase the monetary risk for the bettor and the opponents.
- ***Check/Call***: Make the smallest bet required to stay in the hand (which may be nothing).
- ***Fold***: Take no further part in the proceedings of the hand.

These basic actions are an essential staple of all Poker games. It is the underlying strategy behind the decision-making process of a player that makes the game of Poker arguably one of the most skilful card games in the world. The complexity of Poker results largely from the fact that the only information available to a player of the game's state is that of their card(s) held, the community cards, and that of any past actions the opponents have made.

7.1.1 A Typical Poker Hand

A 'dealer button' denotes which player is the last to play in a hand. In most games the button would be held by the player that deals the cards, but in casino games the dealer never plays. The two players to the left of the dealer post the 'small blind' and 'big blind' respectively, which is a predetermined amount of money (where the big blind is generally double that of the small blind) primarily so that an amount of money is available to win from the beginning of the hand.

Each player is dealt (face down) two cards known as 'hole' cards and the first (pre-flop) round of betting begins, starting with the first player to the left of the 'big blind'. Each player may wager a (in our case, limited) amount of money which is added to the

'pot', or choose to fold out of the hand. Once all players have matched one another's bets (or a winner is declared due to all other opponents folding), three cards known as the 'flop' are dealt face-up as community cards. Another betting round commences, after which a fourth community card is revealed (the 'turn'). Another community card (the 'river') is revealed after another round of betting, after which a final round of betting commences.

The winner is determined by the player that can use any combination of their two hole cards and five community cards to construct the strongest five-card hand. The hand types available to each player are as follows (in order of strength):

- **Royal Flush:** The best possible hand consisting of an Ace, King, Queen, Jack and 10, all within the same suit.
- **Straight Flush:** A five card sequence of all the same suit (e.g. 3, 4, 5, 6, and 7 of spades).
- **Four of a kind:** All four suited cards of equal value (e.g. 5 of spades, 5 of hearts, 5 of clubs and 5 of diamonds).
- **Full House:** Three cards of equal value, with the remaining two cards also of equal value (e.g. King, King, King, Ace, Ace).
- **Flush:** All five cards in the hand are of the same suit.
- **Straight:** A five card sequence, but not all cards are of the same suit.
- **Three of a kind:** Three cards of equal value (e.g. 5 of spades, 5 of hearts and 5 of clubs).

- **Two Pair:** Two separate pairs of cards (e.g. King, King, Ace, Ace).
- **Pair:** A pair of equally-valued cards (e.g. King, King,).
- **High Card:** Given none of the previous hands, a hand valued by the highest card held (e.g. King).

The strength of a hand is inversely proportional to the probability of the hand appearing for a given player. Of a possible C(52, 5) = 2,598,960 card combinations, the probability of receiving a 'High Card' hand is approximately 50.1% in comparison to the 0.000154% of that of a 'Royal Flush'.

7.1.2 Tournament Format

We investigate our approaches using a 'Heads Up' (two player) limit game, which involves bets of limited size. In our experiments we use a 5-chip small blind, and a 10 chip big blind, with all bets within each round limited to a maximum of 40 chips per player. Each player is dealt two cards (known as hole cards) which are kept hidden to all other players. A betting round takes place, where each player must decide whether to fold, check/call, or bet/raise – a bet or raise costs 10 chips. The winner of the hand is the player with the highest valued 5-card hand at the showdown (using any of his own hole cards and the community cards), or the last player left if all opponents fold.

7.2 The AI Players

In creating simple AI opponents, we decide to follow the Loose-Aggressive, Loose-Passive, Tight-Aggressive and Tight-Passive make-up of the previous, One-Card,

experiments highlighted in Chapter 3, Section 3.2.1. Our distinct style players retain the same play style as displayed in Figure 4.1 and Table 4.1.

7.2.1 Evolving an ANN-Controlled Opponent

In Chapter 3, section 4.2.3, "Anti-Players" are created as a 'nemesis' to each of the LA, LP, TA, and TP players. (α, β) pairs were tested in 0.1 increments for $0 \leq \alpha \leq \beta \leq 1$ to determine the best (α', β') pair against each opponent style. Although our previous agent's approach involved a dynamic means of learning to approach an opponent, the strictly static nature of the agent's response renders it unable to use more complex tactics such as check-raises, for example. Our motivation in this chapter is to investigate the potential for evolving a player to be able to develop complex strategic behaviours. In this chapter, we once again evolve players using a C#.NET implementation of NEAT, SharpNEAT (Lockett et al., 2007). Our reasoning behind using the NEAT algorithm is due to the successes of NEAT's topological and weight evolution in finding a suitable network structure for various problems, including game-playing agent control and pole-balancing experiments (Stanley & Miikkulainen, 2002) (Stanley et al., 2005) (Stanley & Miikkulainen, 2002) (Lockett & Miikkulainen, 2008). This choice is further discussed previously in chapter 5, section 5.2.1. For the construction of an 'anti-style' player, we cannot necessarily rely upon a two-parameter design within the realm of full-scale Texas Hold'em. This difficulty is particularly compounded through the use of several betting rounds instead of the previously described game where a there is only a single betting round. The use of α and ß is still applicable to the

application of Texas Hold'em play, but only when considering the separate rounds, and how the probability of winning the hand adjusts based upon the new information yielded by the change of hand status due to the flop, turn and river cards respectively. This in turn makes the calculation of win probability much more difficult, especially due to the expanded number of cards (and suits) compared to the single-card game. Taking into account the use of separate betting rounds, the the design of our network structure has been adjusted such that we use this data as input into the network and can be seen in Figure 7.1.

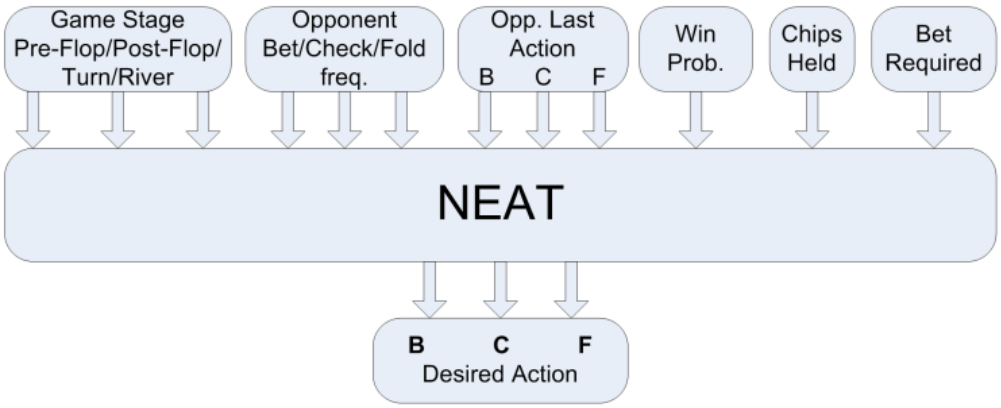


Figure 7.1: Design of player network

The 'Game Stage' inputs are three binary inputs that are all set to 0 if the game is in the Pre-Flop stage, or set to 1 (independently) if the betting round is one of the three further betting rounds. The betting frequency inputs input the frequency at which each of the given actions (bet/check/fold) have been performed by the opponent, normalized over the betting behaviour of the given opponent over the previous 50 hands played. The 'Last Opponent Action' inputs translate the last action of the opponent into binary. The 'Win Probability' input gives the probability of winning the

hand – these use a pre-calculated collection of probabilities from 'Marv742's Flop Tables' (available from the university of Alberta's games group webpage), which is a roll-out of all turn and river combinations for a 7 card hand thus determining the hand odds. This collection of values is split into win probabilities covering various numbers of opposing players, although the win probability can be ascertained from the probability of winning in a 'heads up' tournament, *p*. When given *n* opponents, the probability of winning P*(win)* becomes p^n. We take the collection of pre-calculated values, which are then loaded into a lookup table in memory (values in the table are accessed using a non-suit specific key derived from the current hole cards and the current flop). From our implementation of the lookup table, we can subsequently access the win probability values during the game for each player. The benefit of this table is that on-flop win probabilities can be attained within milliseconds rather than the one-tenth of a second typically used to calculate the probability 'on the fly'. Win probabilities for later betting rounds are calculated using a C# implementation of 'Cactus Kev's Poker Hand Evaluator' by Kevin Suffecool, a technique praised by the members of the University of Alberta Poker group for its speed in hand evaluation (Suffecool, 2006).

The 'Chips Held' input represents the ratio of chips the player has in comparison to the number of chips available at the table (including all opponent-held chips). The final input is a binary input that tells the network if any bet is required by the player to stay in the hand. The outputs of the network represent the action to be taken, the action

represented by the largest numerical output is the action performed. Ties between output values are essentially impossible, but as we have seen in the forthcoming experiments the evolved network generally makes a clear choice for the action to be taken.

7.3 Experimental Results

The evolution of an ANN-controlled player can be seen in Figure 7.2 through Figure 7.5. All experiments are run on a quad-core Intel Xeon X5460 running at 3.16 GHz with 4GB RAM using C#.NET 3.5 running under Windows Server 2003 R2. In these experiments the fitness of each of the solutions in a population is represented by *c*, the number of chips won by a player over 300 hands – This limitation is similar to the reasoning for using a tournament-based approach in Chapter 4, Section 4.1. To run entire tournaments is too computationally expensive (irrespective of using Marv747's Flop Tables as a lookup in memory) and limiting the evolution to a number of hands rather than tournaments is the only feasible compromise. Each player starts with a potentially infinite number of chips, and aims to make the most profit after 300 hands. Each hand is played with a bet limit per round – each bet costs five chips, and up to four consecutive bets can be made per betting round. Each player leaves an Ante of 10 chips for the big blind, and five for the small blind. We use a population size of 100, a node-addition probability of 0.005, and a node-connection addition probability of 0.01 (Stanley & Miikkulainen, 2002). The node addition probability represents the probability that a new node will be added to the network and is limited such that

several generations may be required to adjust to the addition of the new node (and for any connections added between the new node and others). Similarly, too great an addition frequency may reduce the effectiveness of the network in making effective use of the nodes currently in the network. The connection addition probability represents the probability of adding a new connection between any two nodes of the network. These are indicative of the mutative stage of the evolution. There are further capabilities within NEAT to *destroy* connections and nodes, but have not been enabled here, as poorly-performing solutions are generally evolved out of the genome, lessening the need for such destructive measures on potentially promising solutions (Stanley et al., 2005). Each of the subsequent graphs has been averaged over 5 individual runs; note that the 'best fitness' in these graphs represents the average best fitness over the five runs (each of which taking approximately 200 hours apiece), and that these results have been run for 500 generations. The confidence interval is such that over each of the individual runs in the following results, the difference between the performance of the best performing agents (after evolution) was no more than 100 chips won. It should be noted that the 'average' fitness shown on these graphs is an indication of the overall average of the entire population at the time of evolution, and as such the average of *best* performance is indeed the 'best' indicator of overall evolutionary performance.

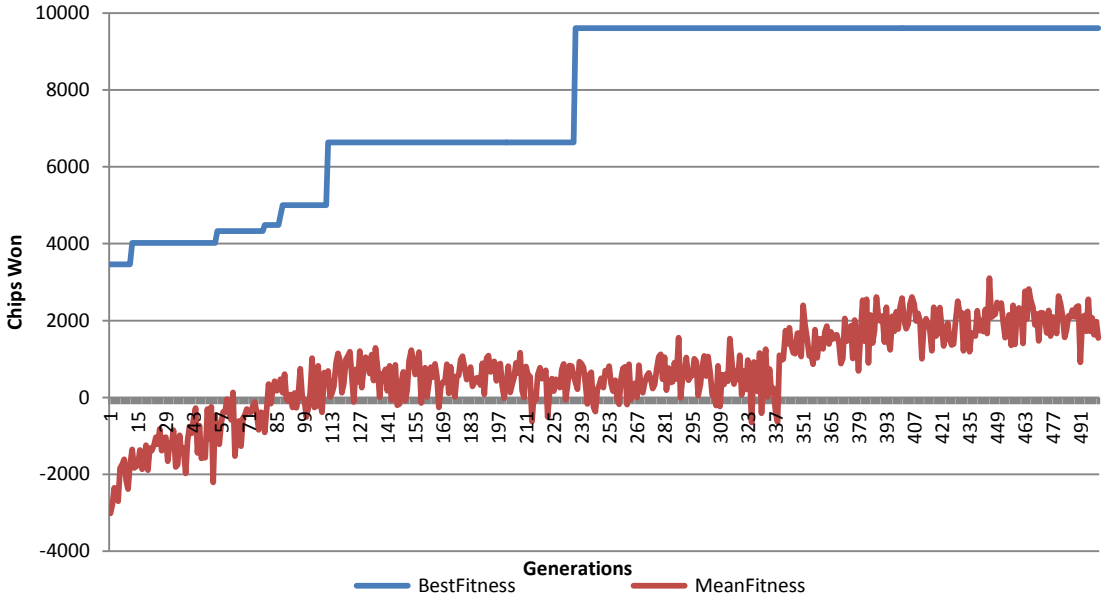


Figure 7.2: Evolution of an ANN against a Loose Aggressive Player

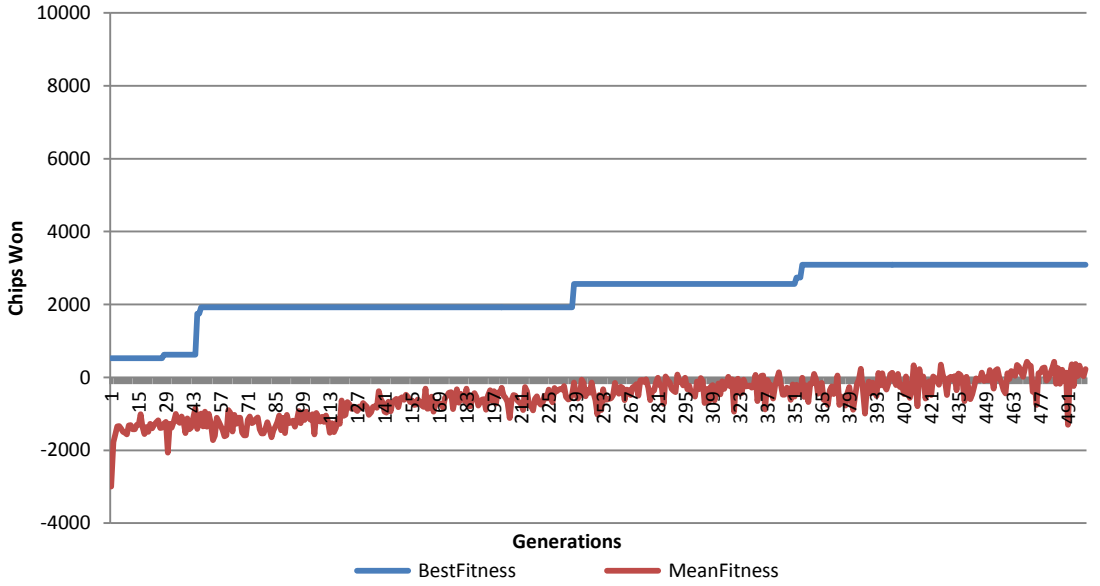


Figure 7.3: Evolution of an ANN against a Loose Passive Player

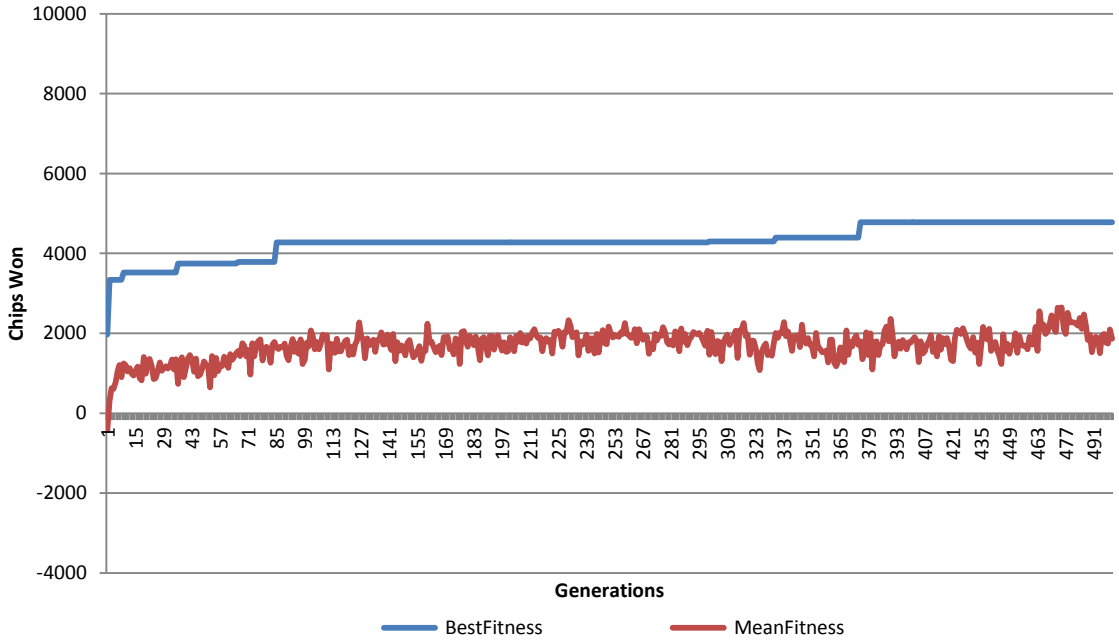


Figure 7.4: Evolution of an ANN against a Tight Aggressive Player

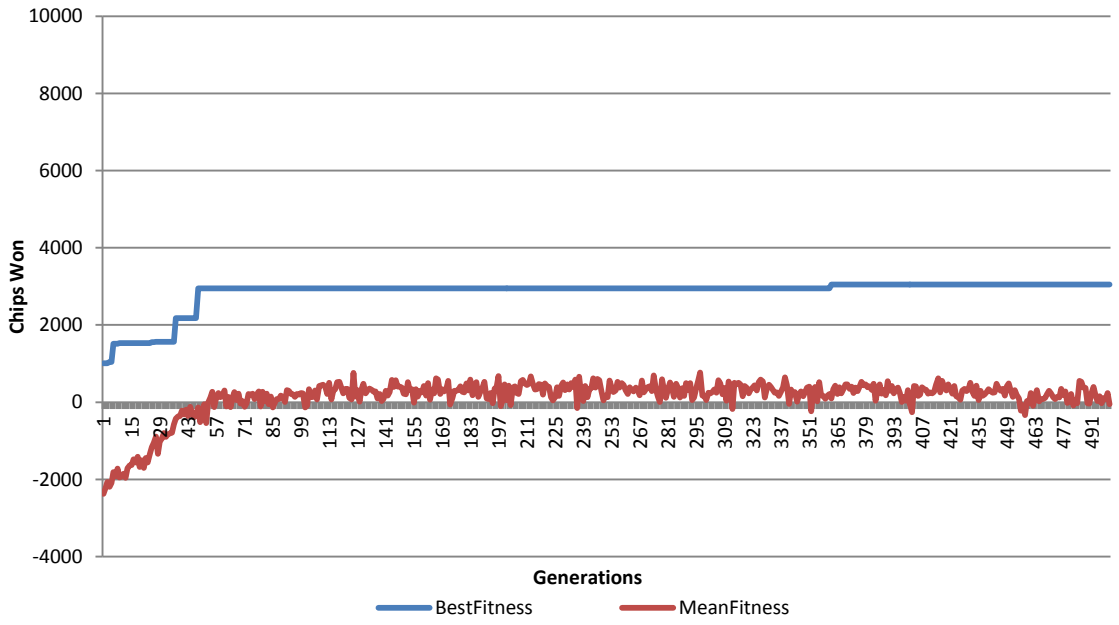


Figure 7.5: Evolution of an ANN against a Tight Passive Player

As can be seen from Figure 7.2 to Figure 7.5, the ANN evolves to a stage where it is able to gain a large chip lead over each type of opponent, the only particularly *large* win being the excessive lead our evolution produces over a Loose Aggressive player in

Figure 7.2. This is mainly due to the fact that the LA player will stay in many hands betting aggressively, therefore putting more chips into the pot thus leading to the large lead by an LA-exploitative player. The performance shown in Figure 7.3 and Figure 7.5 is indicative of the limited amount of money placed into the pot by similarly passive players. As such, calling a bet rather than re-raising will yield smaller pots and a smaller profit over the 300 hands played. A similar case can be noted with Figure 6.4, where the tight nature of the opponent playing limited hands yields fewer 'large' pot wins, and subsequently a lower overall profit – drastically in comparison with the success against Figure 7.2's Loose approach.

Given that a NEAT-evolved player is capable of defeating a single individual style, we aim to evolve a player capable of defeating all four types of opponent in order to reduce the necessity of selection between individual strategies; Figure 7.6 shows the evolution of a player continuing to use the inputs described in Figure 7.1. The fitness determination between evolutionary steps requires that each candidate solution play 300 hands against each opponent in every generation, such that they will play 300 hands against an LA opponent, then 300 hands against an LP opponent, and so on - as such each evolved agent is required to play 1200 hands per generation. The fitness *c* of each solution is again the number of chips won against all opponent styles, this time divided by four to average over each opponent style.

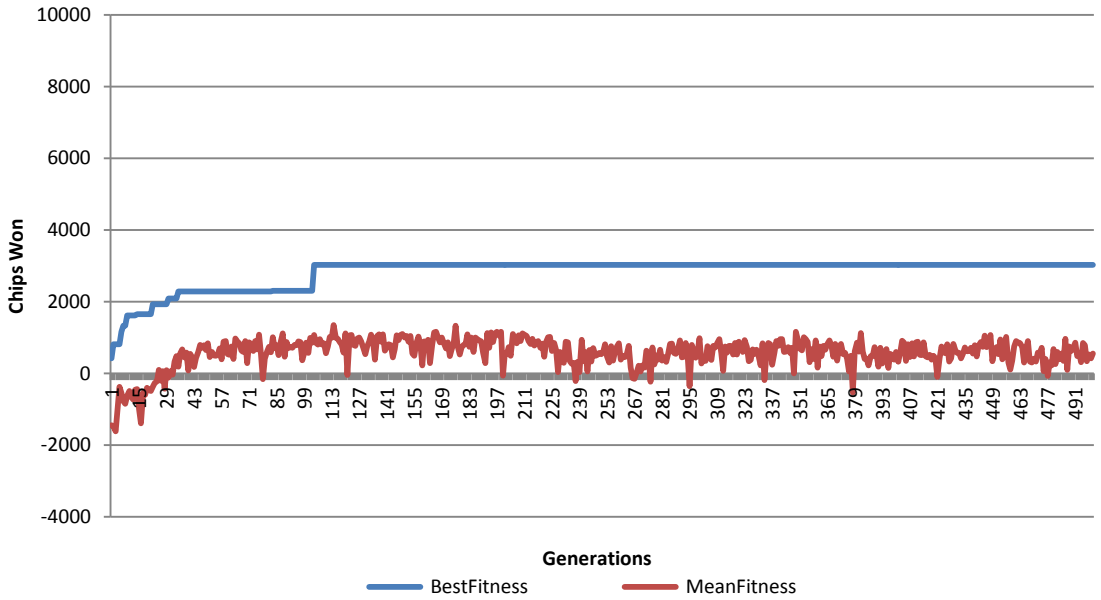


Figure 7.6: Evolution of an ANN against all four player styles.

7.4 Bayesian Analysis

It is noted that the best evolved player average reaches a maximum average success rate of around 3000 chips over 300 hands. If we consider Figure 7.2 to Figure 7.5, we can tell that our players should potentially be able to gain a greater profit than this against these opponents. An opponent model can aid in representing the individual nature of an adversary, and as such could aid in the correct selection of an appropriate reaction to each opponent. Figure 7.7 shows the structure of our proposed network design.

Previous chapters (Chapter 3, Chapter 4 and Chapter 5) have emphasized that Bayes' rule can be a powerful learning approach that can analyse the past play information of an opponent, and determine useful information about each opponent's respective

playing style, as well as showing that an opponent model can aid the overall quality of an evolved ANN against multiple styles of opponent.

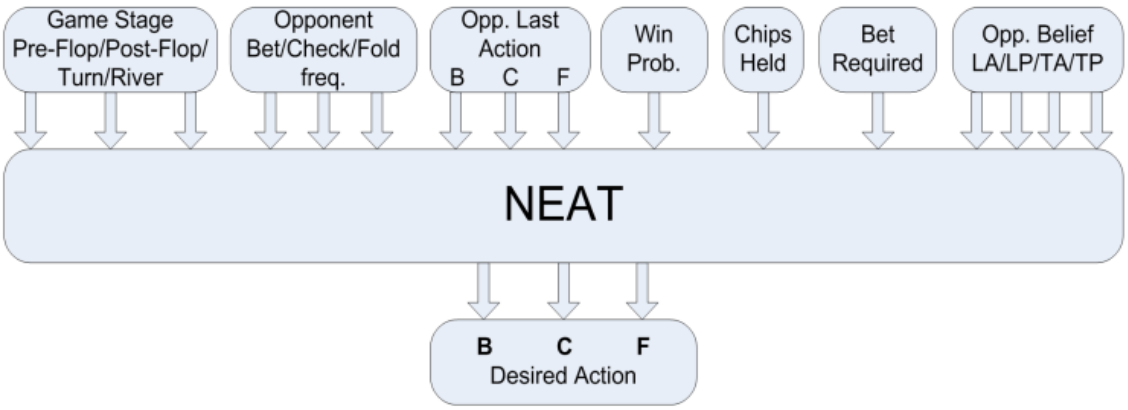


Figure 7.7: Design of proposed player

The usage of Bayesian probabilities to model uncertainties is popular in relation to imperfect information games such as Poker (Burns, 2006).

The probabilities in Table 7.1 represent an *a priori* P(*B* | *A*) = P(*Action* | *Strategy*) set of probabilities which were evaluated by analysing the past actions of players of style *A* over a 10,000 hands (2,500 hands were played against each separate style and the values averaged over all hands played). P(*A*) is the prior belief of an opponent's play style, and as such is set to 0.25 initially as all opponent styles are assumed equally likely at the start of a game. Bayes' rule updates the initial probability of player style belief such that P(*A* | *B*) for the current iteration becomes P(*A*) for the next action analysed (i.e. our belief of style P(*A*) is the result of P(*A* | *B*) for the last action analysed). P(*B*) is represented by the summation of P(*B* | *A*)P(*A*) for all possible *A* (represented by *a*), and is used as a normalising constant.

	FOLD	CHECK /CALL	BET /RAISE
LA	0.03	0.07	0.90
LP	0.01	0.94	0.05
TA	0.25	0.55	0.20
TP	0.25	0.70	0.05

Table 7.1: Action probabilities for each opponent style.

The Bayesian Observation Window

As the Style belief of our Bayesian model converges towards 1 (given a static opponent style; bluffing or deceptive play varies these styles and therefore probabilistic convergence may be limited), the capability of our model to adapt to different styles becomes compromised. To exemplify this, we use the same Bayesian probabilistic update as shown in Algorithm 4.1. upon a set of recorded actions by a Loose Aggressive player (i.e. a primarily betting opponent). If we observe Figure 7.8, we can see that the probabilities for all other styles of opponent approach zero (just as the LA belief approaches 1), and given how Equation 1 (and the code from Figure 4.2) updates its probabilistic beliefs the probabilities would potentially take far too long to recover from over-convergence and accurately represent a change in opponent style. This is due to the *a priori* belief being utilised as a multiplying factor therefore compounding the minimization of opponent style belief over time. Since the relative behaviours represented in Table 7.1 are not drastically different from one another, the change in belief is more subtle than it would be if there were only two styles of player

where style one played with P(BET)≈0.9, and the P(FOLD)≈0.9 (such a definition would lead to a very swift change in belief given random bet/fold actions).

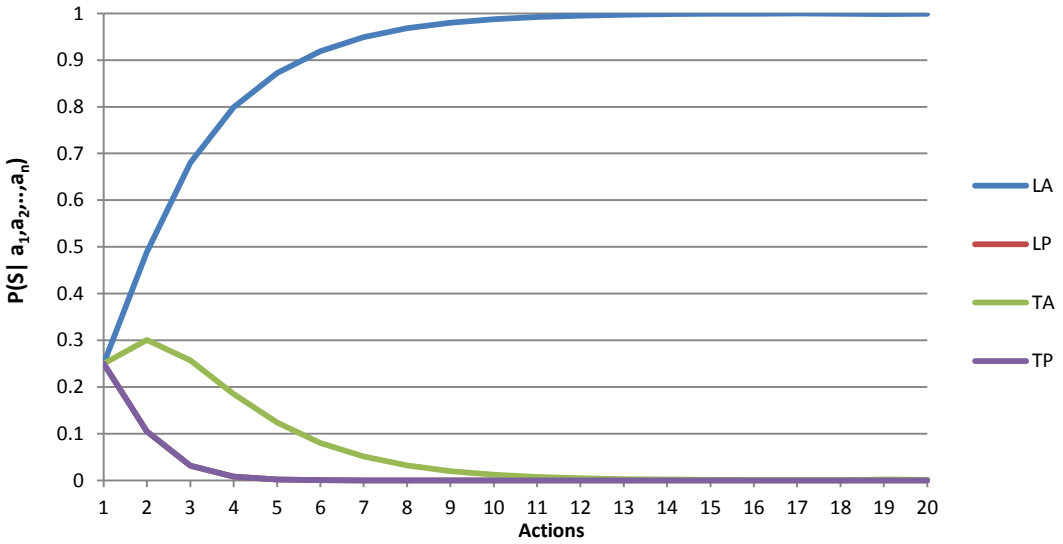


Figure 7.8: Probability convergence of the Bayesian model given a set of LA-type actions.

To counteract this issue, we use a *windowed* analysis of actions, such that only the past *n* actions of an opponent are accounted for, yielding a faster recognition of opponent style change. Whenever our agent is requested to perform an action, all four beliefs of opponent style are set to the default value of 0.25, and the past *n* actions are used to determine opponent style. For all further experiments in this chapter, *n* shall be set to a value of 8. The effect of a limited Bayesian observation window in comparison to our normal Bayesian model can be seen in Figure 7.9, where we observe a tight, passive player changing style to a Loose Aggressive one over a period of 30 actions. The Bayesian probabilities are updated as in Algorithm 4.1, against a sample of the actions made by a Tight Passive player moving to the style of a Loose Aggressive one.

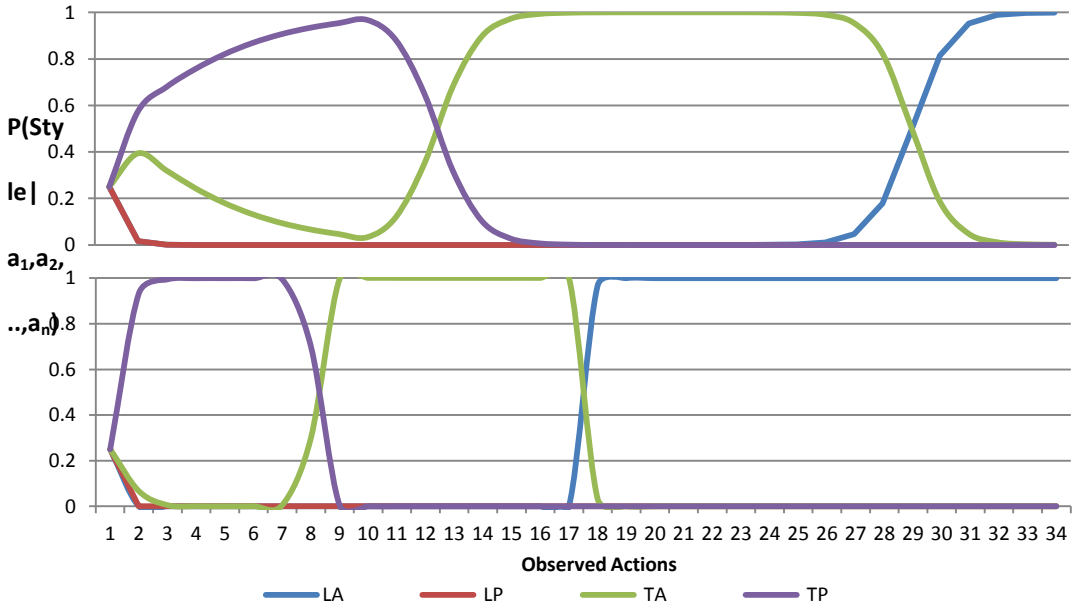


Figure 7.9: Comparison of probability convergence using standard Bayes' rule (top), and windowed analysis (bottom) The opponent changes style from Tight Passive to Loose Aggressive at action number 7.

As we can see, the analysis briefly traverses through the Tight Aggressive label whilst approaching a Loose Aggressive belief. This is understandable due to the frequently folding nature of a TP player as well as the frequently betting nature of an LA player, the central analysis containing a distribution of folding and betting actions which the TA player represents. In the top graph we note that the recovery of our analysis from TP to LA is slow due to over-convergence, whereas the windowed approach quickly resolves towards an LA belief although the mix of folding and betting inevitably yields a Tight Aggressive analysis, this is quickly overcome. The per-style Bayesian beliefs (with a limited observation window) are subsequently used as four further inputs to augment the evolution of our player. Figure 7.10 displays the effect that opponent model augmentation has upon the evolution. The experimental conditions are the same as those described for Figure 6.6 in Section 6.3; The fitness determination

between evolutionary steps requires that each candidate solution play 300 hands against each opponent in every generation, therefore 1200 hands per generation. The fitness *c* of each solution is again the average total of chips won against all opponent styles.

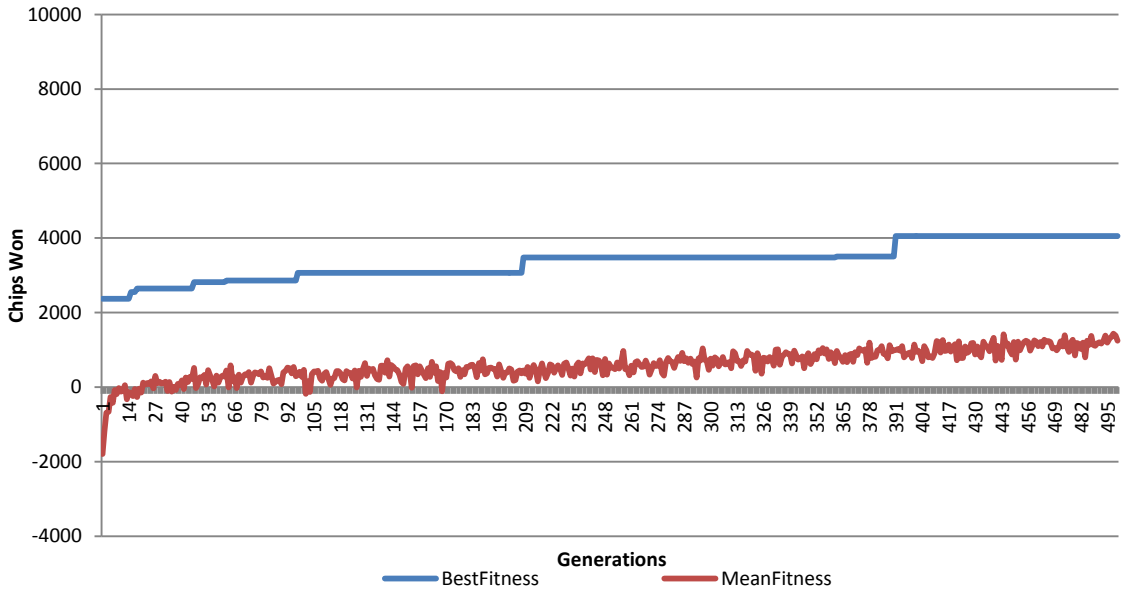


Figure 7.10: Evolution of a Bayesian Model-augmented ANN against all opponent styles.

We can see that the network evolves to a solution which reaches a best profit of 4000 chips after 300 hands, a 33% improvement over the non-Bayesian ANN player in Figure 7.6. It appears that, through observation of the average fitness, greater improvement could be attained through further evolution, as the still increasing gradient (of the average population fitness) suggests towards the end of our 500 generations – a stark contrast to the stagnant average fitness in Figure 7.6.

7.5 Observing the Evolved Performance

The inclusion of an opponent model appears to aid both the evolution and decision process of the agent. This appears to be due to the ANN's ability to utilise the opponent model's separation of opponent types in order to determine a reasonable course of actions against the opponent(s). In order to test this theory, we observe the performance of the overall 'best' performing networks for both experiments (from Figure 7.6 and Figure 7.10) at the end of evolution. We play our best networks against each static style of opponent over a period of 1000 hands, recording the hand-to-hand performance of each player against the original four simple players. We once again use the conditions described in Section 7.3, where an infinite number of chips is assumed, with players limited to four bets/raises/re-raises per round. These results can be seen in Figure 7.11 through Figure 7.14.

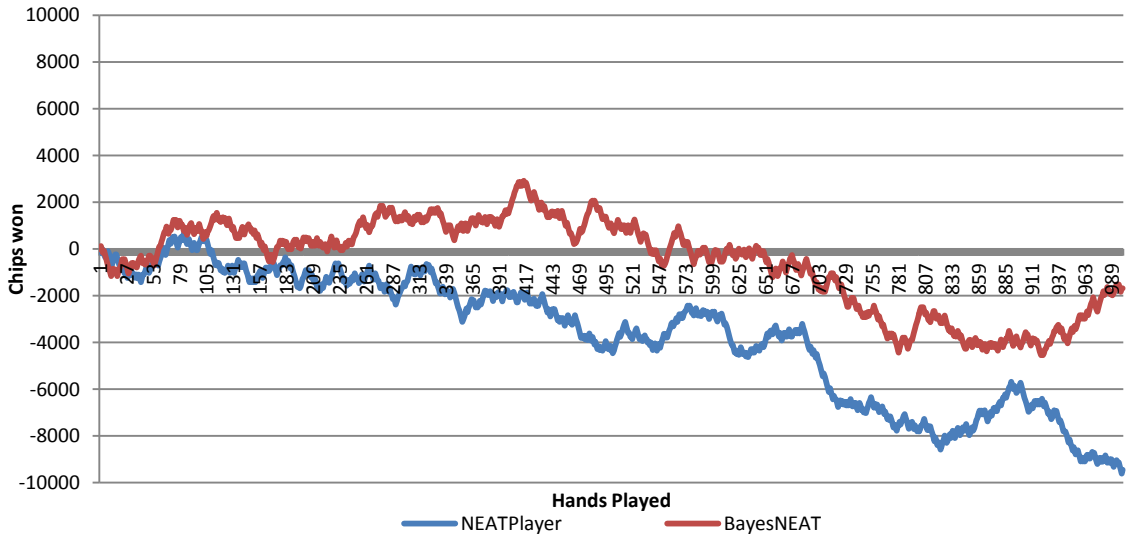


Figure 7.11: Performance of the 'best' evolved ANN against a Loose Aggressive Player.

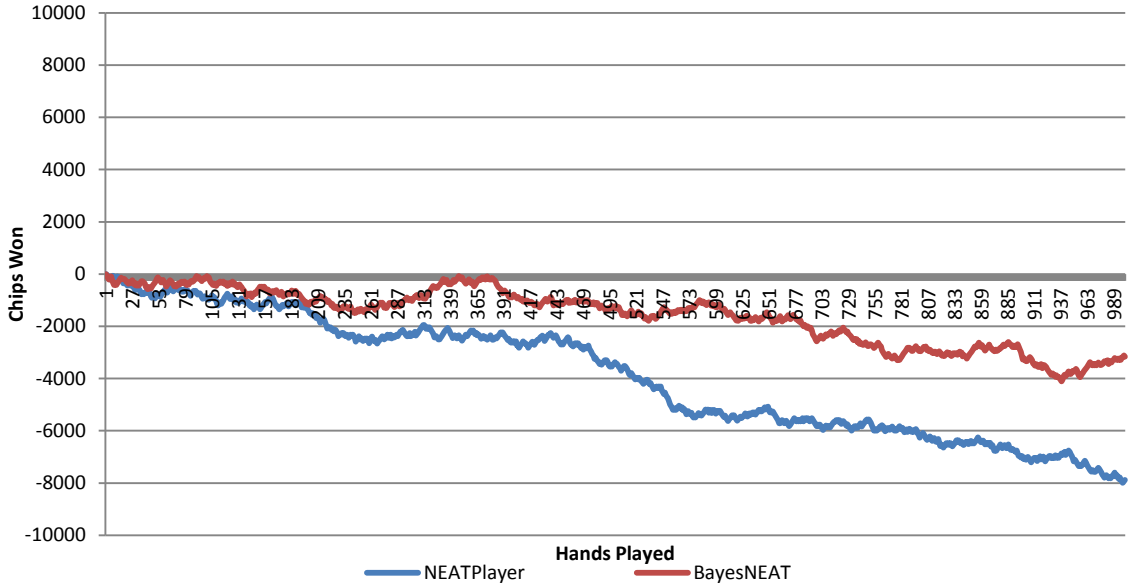


Figure 7.12: Performance of the 'best' evolved ANN against a Loose Passive Player.

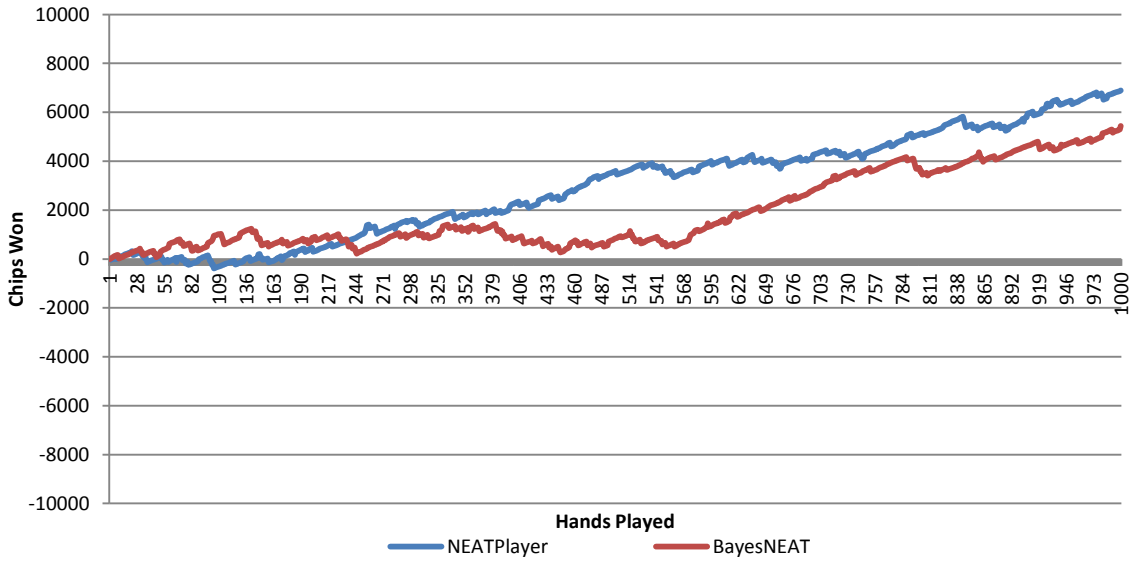


Figure 7.13: Performance of the 'best' evolved ANN against a Tight Aggressive Player.

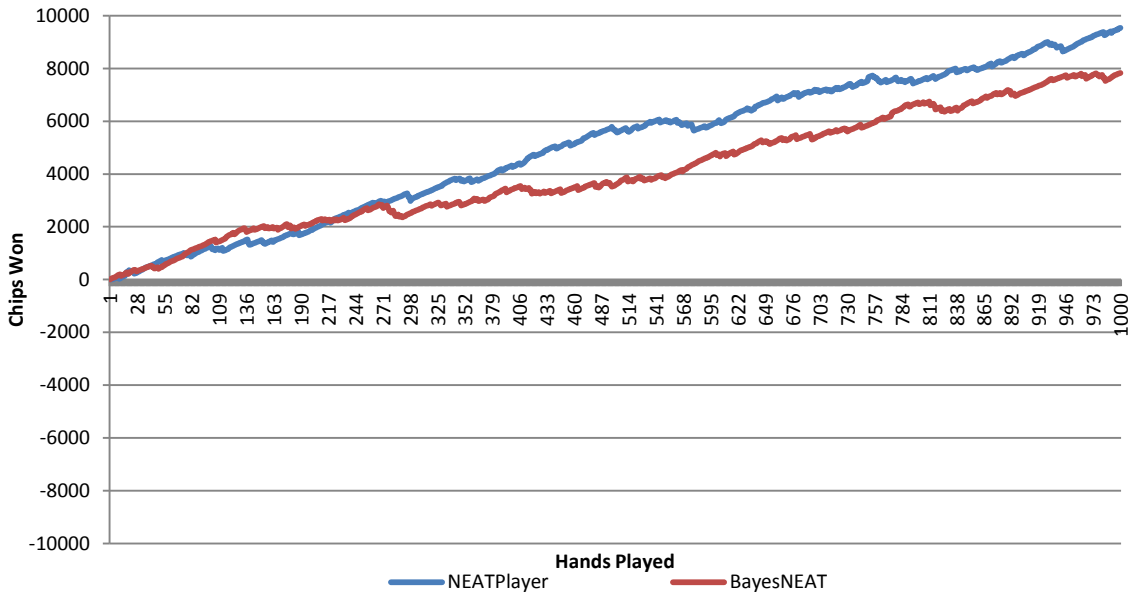


Figure 7.14: Performance of the 'best' evolved ANN against a Tight Passive Player.

Figure 7.13 and Figure 7.14 display a successful performance over the 1000 hands, consistently improving their chip-count over successive hands. Conversely, the results shown in Figure 7.11 and Figure 7.12 actually display *poor* performance. The overall average performance during evolution (Figure 7.6 and Figure 7.10) can be explained through the high values reached against TA and TP players being pulled back down by weak performances against LA and LP players. We can observe that the Bayesian player performs better than the standard NEAT player when the opponent is stronger, but performs slightly weaker against an opponent that is easier to beat. The Bayesian approach does seem to aid the quality of overall performance, but not to an extent within which it encourages a strong performing player. This behaviour is highly contrasted to the evolved performance against Loose Aggressive and Loose Passive styles displayed in Figure 7.2 and Figure 7.3. In the previous examples, a network aimed specifically at each of the opponent styles successfully outplayed the static

agents, but in this instance the evolved solutions appear to only be responsive to tight opponents. The contributing factor to the limited scope of the evolved players' success could be the averaging nature of the fitness function used, which does not promote the improvement of success against opponent styles with which the solution is struggling. Another potential cause of weakness can be related to the construction of our model; as our previous work from chapters 4 and 5 uses a single betting round, we also use a single set of probabilities for our model. Just as Schaeffer's detailing of attributes necessary of a world class Poker player (Billings et al., 2001) include that of being able to understand that the win potential of a hand could depreciate as well as improve, our model needs the ability to identify the changing nature of a player over the different betting rounds – win probabilities adjust given the addition of new community cards, and a two-parameter α-β player's behaviour would switch as such given the new probabilistic information.

7.6 Round-Based Modeling

Considering the multi-round nature of full-scale Texas Hold'em, an individual model per betting round could prove far more useful, as betting behaviour can convey not only the style of an opponent, but also a potential change in the quality of an opponent's hand due to subsequent community cards being revealed. The probabilities in Table 7.2 show the average behaviour of each type of opponent after 10,000 hands (2,500 played against each style of opponent).

		Fold	Check /Call	Bet /Raise
LA	**PreFlop**	0.05	0.05	0.90
	PostFlop	0.05	0.12	0.83
	Turn	0.10	0.15	0.75
	River	0.15	0.25	0.60
LP	**PreFlop**	0.025	0.95	0.025
	PostFlop	0.05	0.90	0.05
	Turn	0.05	0.85	0.10
	River	0.15	0.80	0.05
TA	**PreFlop**	0.42	0.53	0.05
	PostFlop	0.11	0.60	0.29
	Turn	0.11	0.56	0.33
	River	0.05	0.52	0.43
TP	**PreFlop**	0.53	0.42	0.05
	PostFlop	0.13	0.82	0.05
	Turn	0.10	0.80	0.10
	River	0.05	0.80	0.15

Table 7.2: Action probabilities for each opponent style per betting round.

As Table 7.2 shows, the betting behaviour displayed by each type of player varies between betting rounds (as assumed in Section 7.6), in some cases even overlapping with the behaviour of a different style in another round. It appears that in the case of the two Passive players the action-style probabilities actually display a switch of *styles* – For the case of a Loose Passive player, many hands are played, and therefore stayed in due to the behaviour of the Loose Passive player to call and bet or raise. Towards the later betting rounds, as the Loose Passive player is prone to staying in a hand with a potentially poor set of hole cards (and is likely to be unlucky on the flop), the win

probability for an LP player drops dramatically and the α and β values used would dictate that the player becomes characteristically a Tight Passive player – The Loose Aggressive player also appears to adapt in that the increasing weakness of the hand yields a greater probability of folding. Conversely, yet similarly, a Tight Passive player would generally exclusively stay in play with a strong hand in the earlier stages, such that the probability of winning would more likely increase (when in comparison to a Loose Passive players hand) and therefore as the probability increases, the more likely the Tight Passive player is to bet during the post-River round. This yields a greater explanation as to the performance of our evolved agents in Figure 7.11 and Figure 7.12; the drastic change in opponent behaviour leads the opponent model to an incorrect solution as to the style of the opponent which, in turn, affects the performance of play.

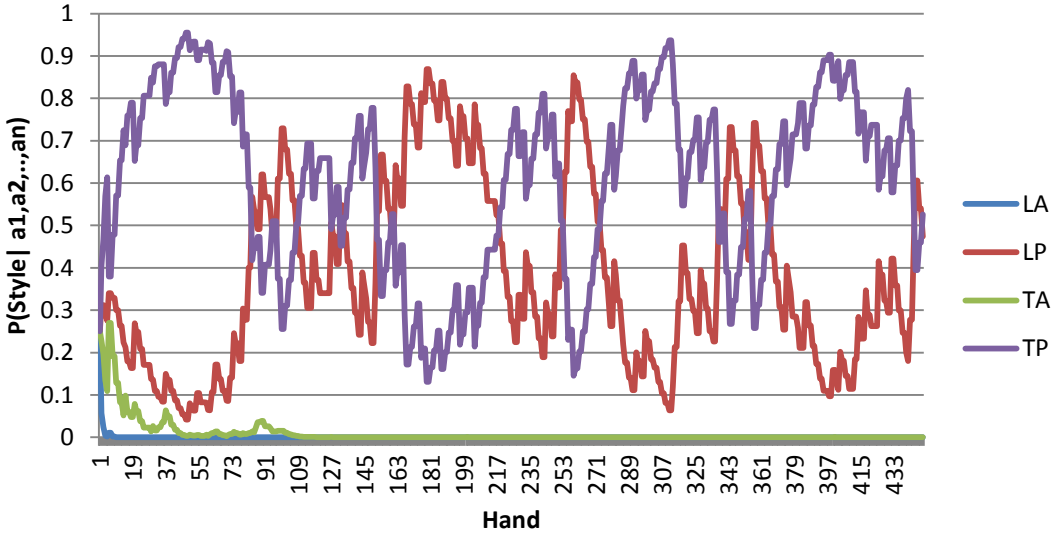


Figure 7.15: Analysis of a Loose Passive player using a single-round Bayesian approach

The effect of this incorrect analysis (as evidenced by Figure 7.15) is that as the Loose Passive player is subsequently believed to be Tight Passive at the end of a hand (in part expedited by the rapidity of change of style belief using the windowed analysis from Section 7.5). This then means that the agent will use this model to respond to a potentially Tight Passive opponent at the start of the next hand, meaning that the Loose play will then be taken by the agent as a display of the opponent holding a good opening pair of hole cards. Subsequent folding by the agent (in the assumption of a tight opponent) will then provide the Loose Passive player with chips, akin to the tactic of 'blind stealing', which is explained further in Chapter 5, Section 5.7. The Bayesian analysis of each style of opponent (given a set of style-cased opponent actions) can be seen in Figure 7.16 through Figure 7.19, which show the difference in convergence speed given the disparate action frequencies presented in Table 7.2. These results further display the importance of heterogeneity in the different Action-Style probabilities in determining opponent type, as discussed in Chapter 4, section 4.2.4.

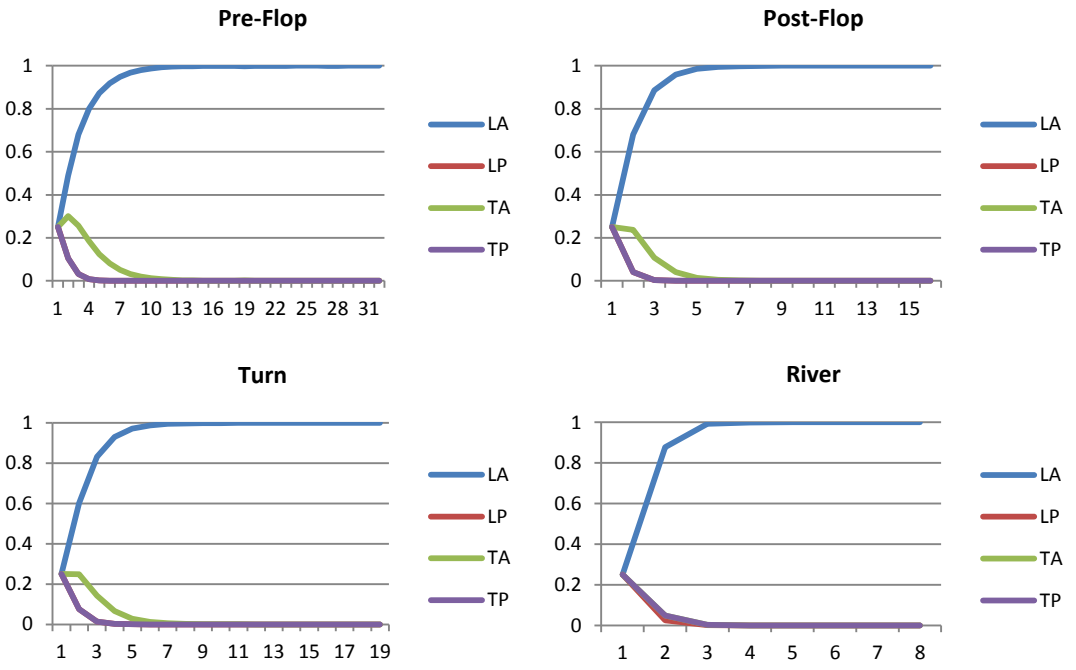


Figure 7.16: Bayesian Multi-Model Analysis Vs. A Loose Aggressive Player

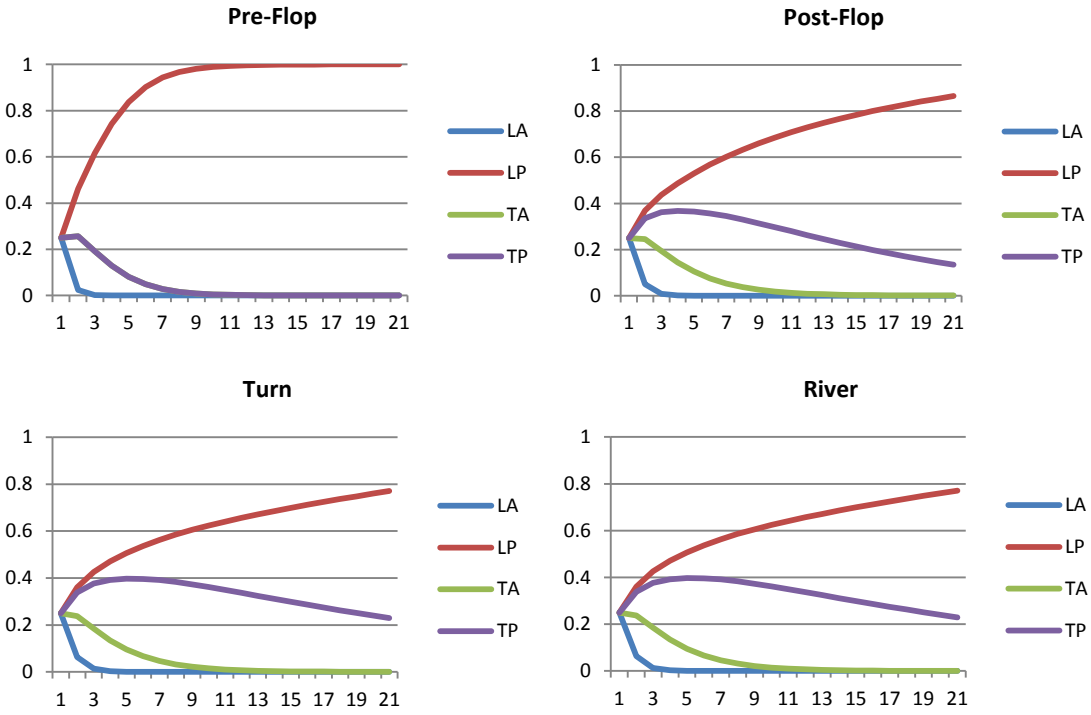


Figure 7.17: Bayesian Multi-Model Analysis Vs. A Loose Passive Player

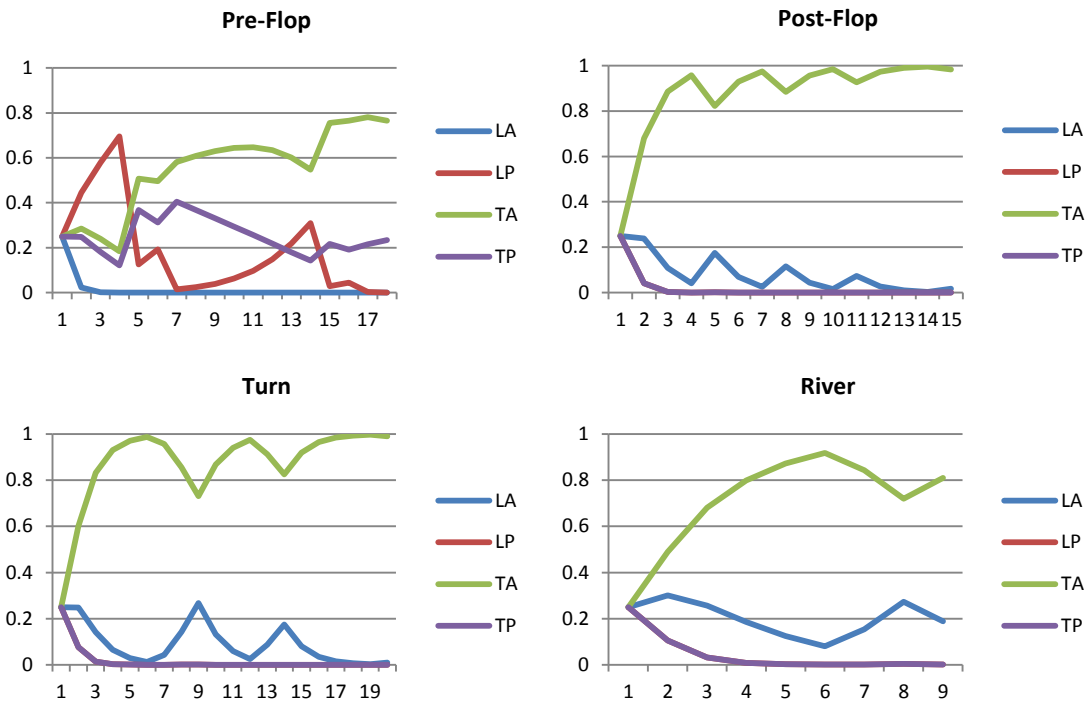


Figure 7.18: Multi Bayesian Model Analysis Vs. A Tight Aggressive Player

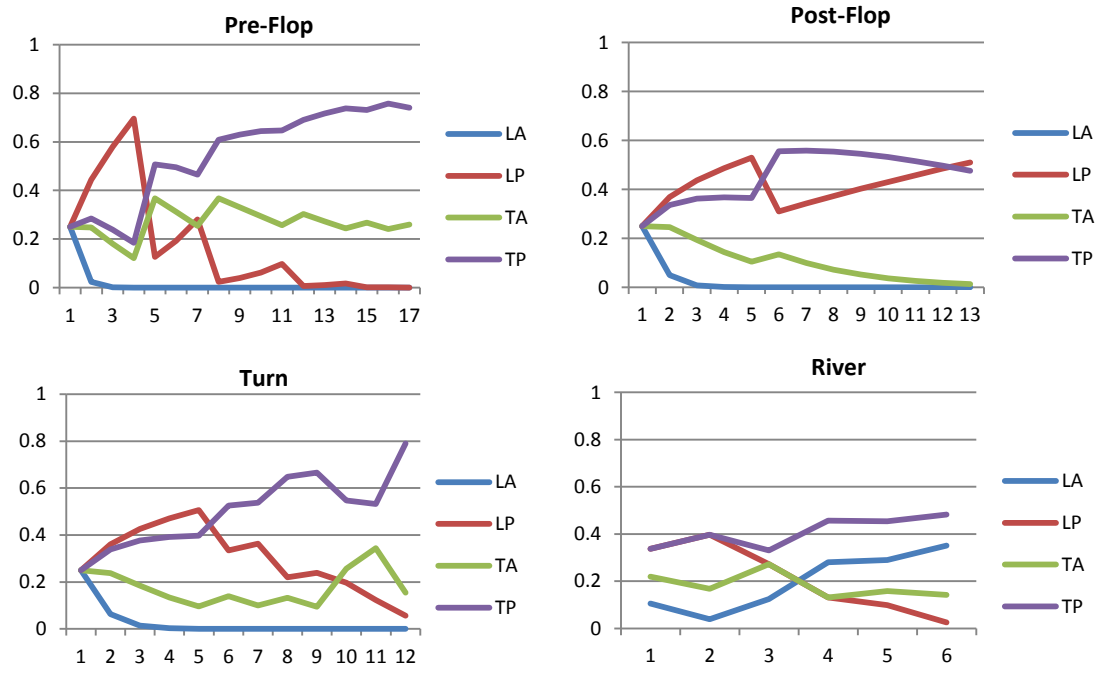


Figure 7.19: Multi Bayesian Model Analysis Vs. A Tight Passive Player

We now evolve an ANN-controlled agent using the same network structure as in Figure 7.10, again requiring each candidate solution to play 1200 hands per generation, 300 against each type of opponent. The fitness *c* of each solution is once again the average total of chips won against all opponent styles. Evolution and evaluation using the same parameters allows us to investigate the effect our separated model has upon the quality of evolution and therefore the evolved solutions. The evolution of this agent can be seen in Figure 7.20, which is averaged over 5 individual runs.

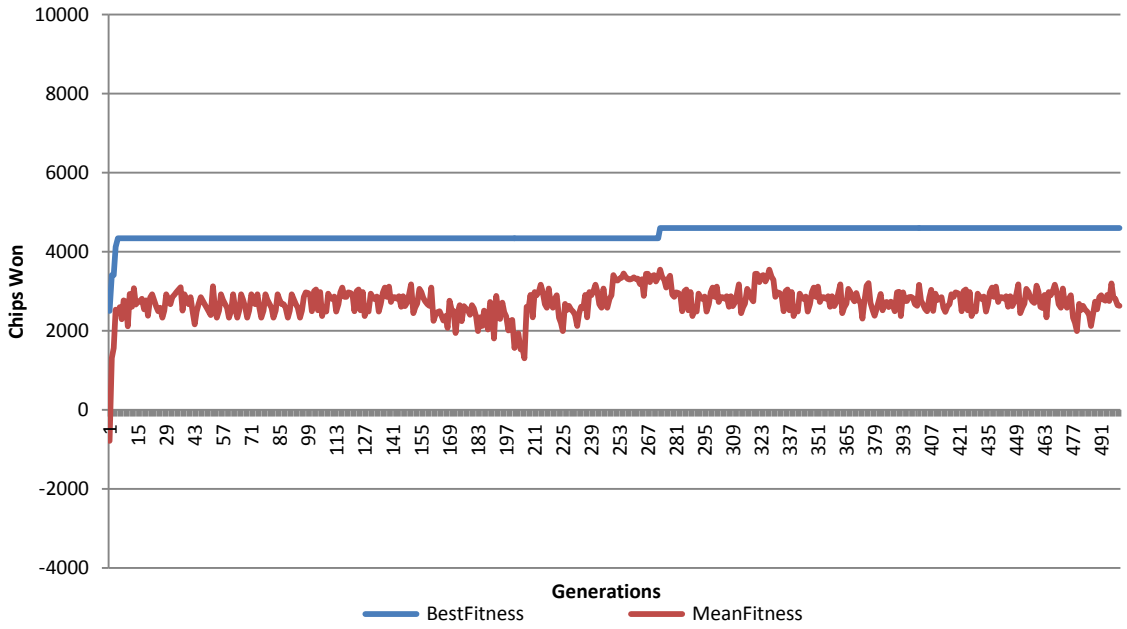


Figure 7.20: Evolution of a multi-round Bayesian player.

The evolution appears to show little improvement upon the evolutionary performance of our standard NEAT or single-round Bayesian approaches; the averaging nature of the fitness function can mean that the performance can be contradictory in nature (as displayed in Figure 7.11 and Figure 7.14), or consistently good. As with Figure 7.11 through Figure 7.14, we play our best networks against each

static style of opponent over a period of 1000 hands, recording the hand-to-hand performance of each player against the original four simple players. We once again use the conditions described in Section 7.3, where an infinite number of chips is assumed, with players limited to four bets/raises/re-raises per round. The performance of our multi-round modeling approach with respect to NEAT is compared to our previous two 'best' players in Figure 7.21 to Figure 7.24.

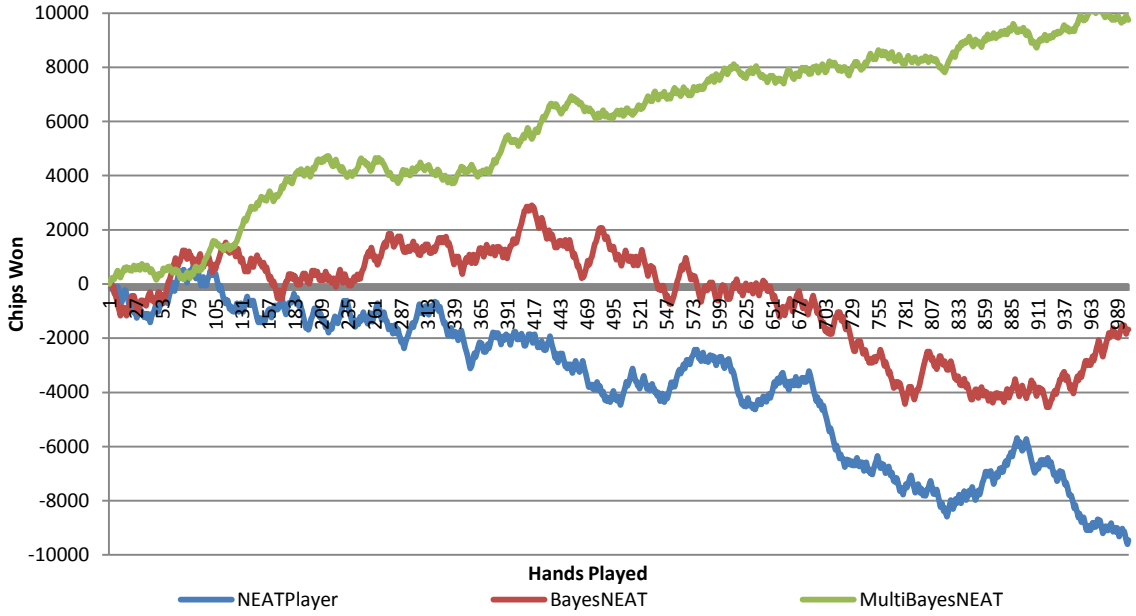


Figure 7.21: Performance of the 'best' evolved ANNs using each strategy against a Loose Aggressive Player.

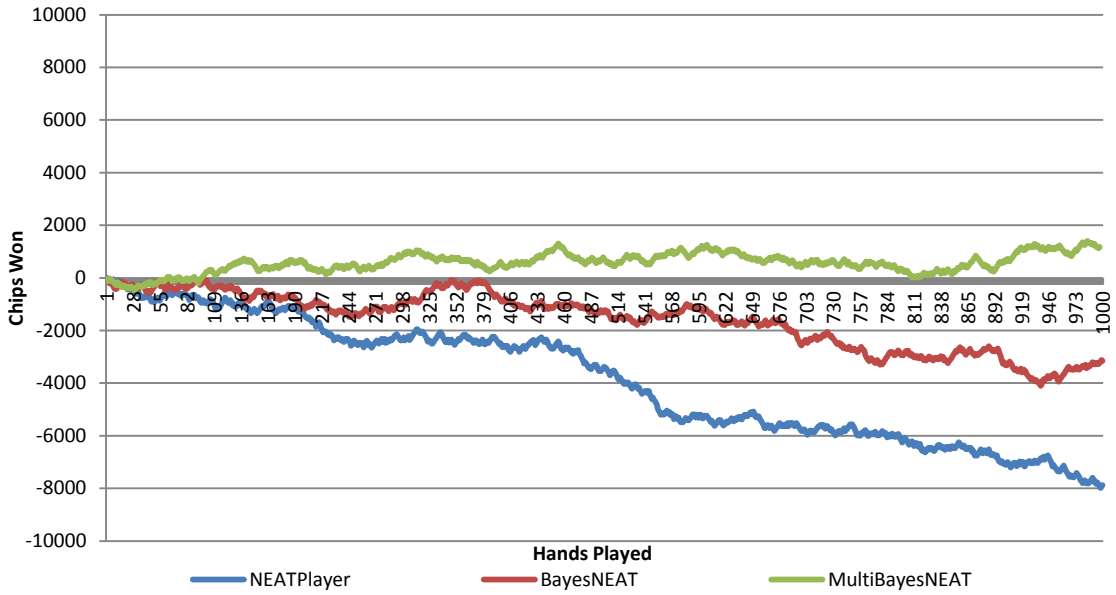


Figure 7.22: Performance of the 'best' evolved ANNs using each strategy against a Loose Passive Player.

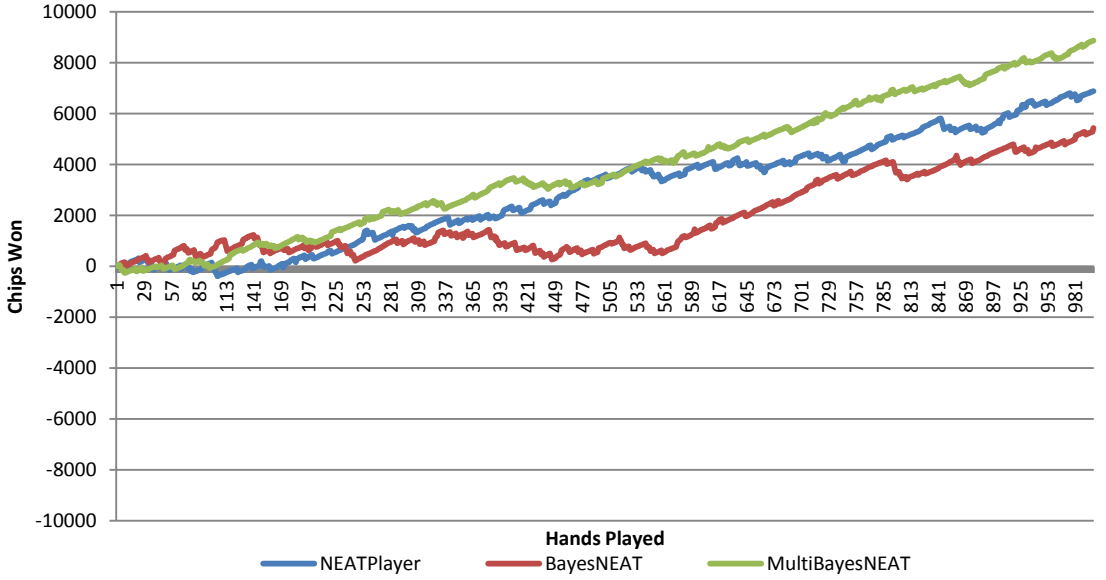


Figure 7.23: Performance of the 'best' evolved ANNs using each strategy against a Tight Aggressive Player.

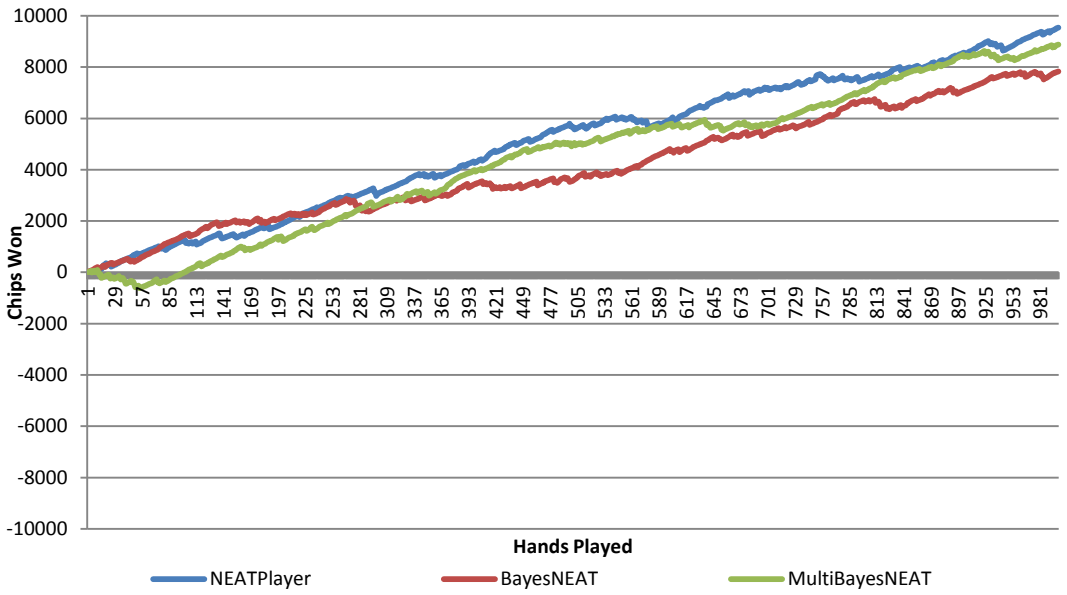


Figure 7.24: Performance of the 'best' evolved ANNs using each strategy against a Tight Passive Player.

As we can see from Figure 7.21 through Figure 7.24, the multiple-round modeling yields stronger play from our evolved agent against all opponents, especially in relation to the Loose Aggressive player. The adapted model is less likely to adjust incorrectly to a Tight style belief given the change in behaviour by the Loose Aggressive player; this adjustment means that the 'bluff stealing' approach discussed earlier is less prevalent due to the agent now remaining in the hand given the lesser likelihood of the opponent holding a strong hand. The performance against a Loose Passive player is still relatively weak. This could be attributed to the behaviour of a Loose-Passive opponent playing in most hands which is quite frequently a preferred tactic in a heads-up situation, and can be considered as playing 'smartly', rather than the Tight Passive approach to playing only 'frugally' (Sklansky, 1992). In this scenario, any hands the Loose Passive player stays in are low-pot situations (due to the passive checking behaviour), and staying in more hands with a low pot can potentially lead to a

'stalemate' as can be seen by the very low winnings displayed in Figure 7.22 over 1000 hands. The performance against the Tight players is comparable to the NEAT-only and Bayes' rule-NEAT approaches, displaying that a multi-round approach is a successful adaptation of the model when dealing with a multiple betting round scenario. The evolved agent adopts a strategy of blind stealing (where the Tight Passive player is liable to fold when facing aggressive play), as well as folding where the Tight player bets and limiting its betting behaviour where moderate to good hole cards are held.

7.7 Response to Bluffing

We modify each of the individual opponent styles that it bluffs with probability p. Upon any decision by the agent, with probability p we change the action performed to that of a betting action; this will gradually adapt the playing style of any agent. For example, we take the evolved ANN from Figure 7.5 (herein referred to ANNti-TP), which is evolved against the Tight Passive player and observe the performance over 300 hands against a Tight Passive opponent that bluffs with probability p. The value of p is adjusted in increments of 0.01 from $p = 0$ to $p = 1.00$. 300 hands are played against a Tight Passive player using each value of p. Figure 7.25 shows the weakness of adopting a static strategy (in this case ANNti-TP) against a dynamically changing opponent.

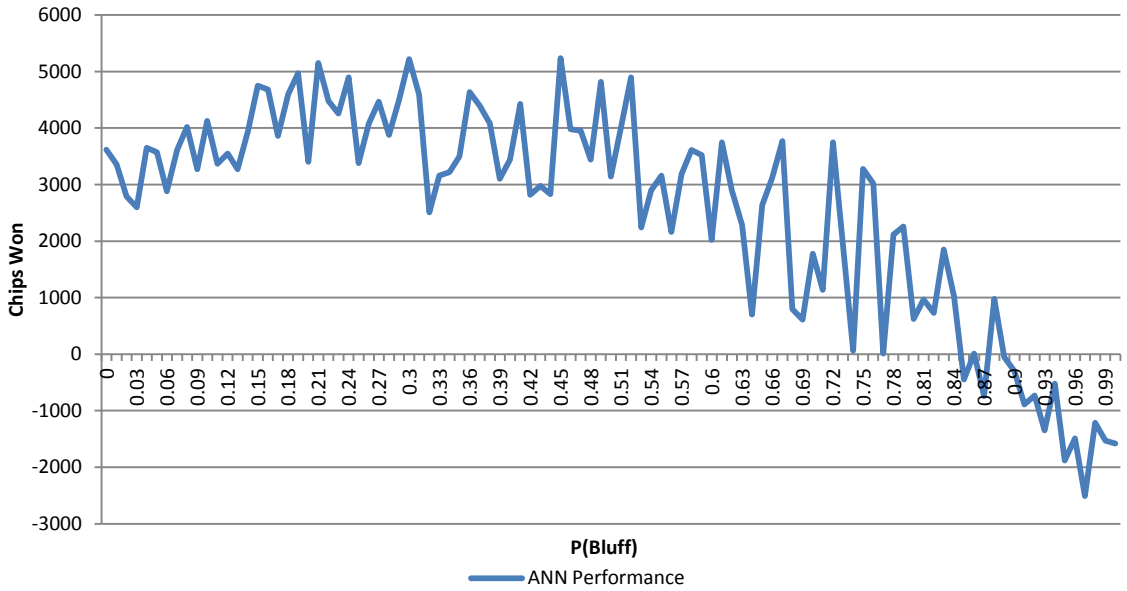


Figure 7.25: Performance of an evolved ANN versus a Tight Passive player with a varied bluffing level.

As we can see the performance of ANNti-TP becomes unstable as the opponent becomes more unlike that which it is evolved against. This is understandable as the experiments from Chapter 4, Section 4.4 have shown that as the probability of bluffing (or betting in general if we consider tight to be synonymous with reserved play, restrained even) increases, the closer a tight player becomes to that of a looser one, where betting is concerned, this approaches the behaviour of a Loose Aggressive opponent. Let us compare the performance from Figure 7.25 with that of the previously evolved Multi-Round Bayesian ANN from Figure 7.20; we use the same Tight Passive player, bluffing with probability *p*, in 300 hands over 0.01 increments of *p*, and examine the effects of bluffing (or indeed strategy adjustment) against a player using the Bayesian opponent analysis displayed in Figure 7.26.

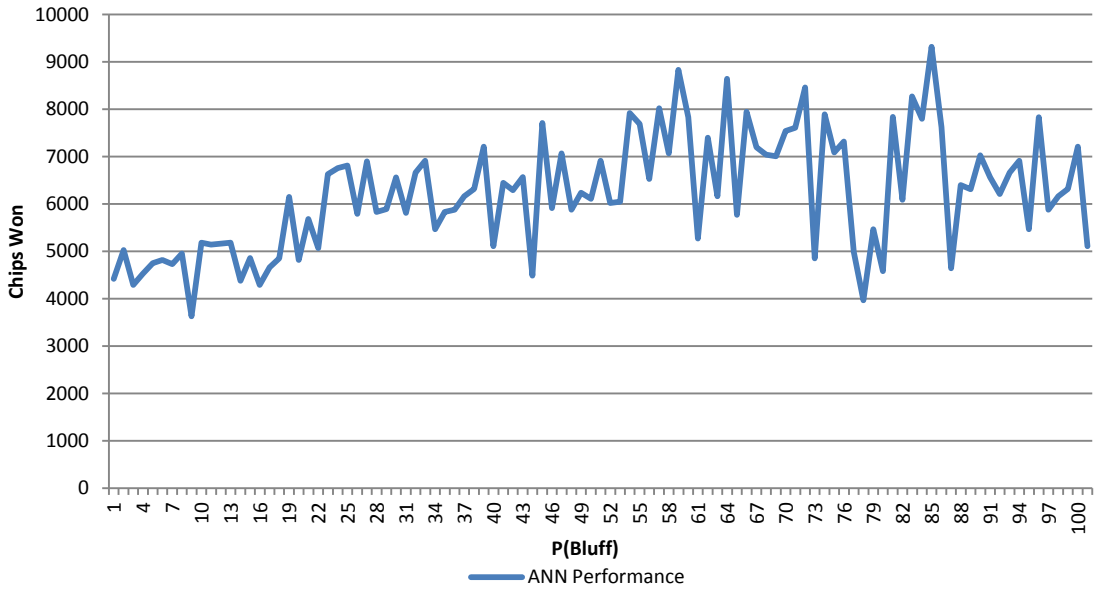


Figure 7.26: Performance of an evolved Multi-Round Bayesian ANN versus a Tight Passive player with a varied bluffing level.

We can see that, irrespective of the inevitable change in opponent style with respect to the altering level of bluffing, a player using the Bayesian model copes suitably well; the *stability* of the performance wavers somewhat towards the greater probability of bluffing (as evidenced around the 0.73-0.90 area). This is understandable as the nature of dealing with an extremely loose player is to call what is presumably a weak hand – the outcome of this would be to mostly call the opponent's bluff which takes a potentially significant risk, therefore the reward can be somewhat erratic.

Our modeling approach aids the agent in its adaptation to an opponent's behaviour, even if the opponent may not necessarily adhere to the strict definition of the individual styles. An assumed opponent style which is loosely affiliated with one of the styles is enough to aid the performance of the agent. As Figure 7.26 displays, the adaptive bluffing nature of the opponent adjusts the initially Tight Passive player to a

style not necessarily defined within the LA-LP-TA-TP bounds, but as the model assumes a style, the agent responds to the *perceived* type. This was also exemplified in the single card game, particularly in Section 5.6 where the resulting populations of co-evolved players were of drastically differing styles, yet the agent remained stable in its playing success.

7.8 Dynamic opponents

In order to test the ability of our player to adjust to the potentially dynamic nature of opponents we test our player (our best single ANN from the experiment portrayed in Figure 7.20) against dynamic opponents. We implement a series of players that transition from one style to another once every 75 hands for a total of 300 hands. Figure 7.27 shows the performance of our agent against a Dynamic player that transitions from LA to LP to TA to TP, the vertical lines denoting each style change.

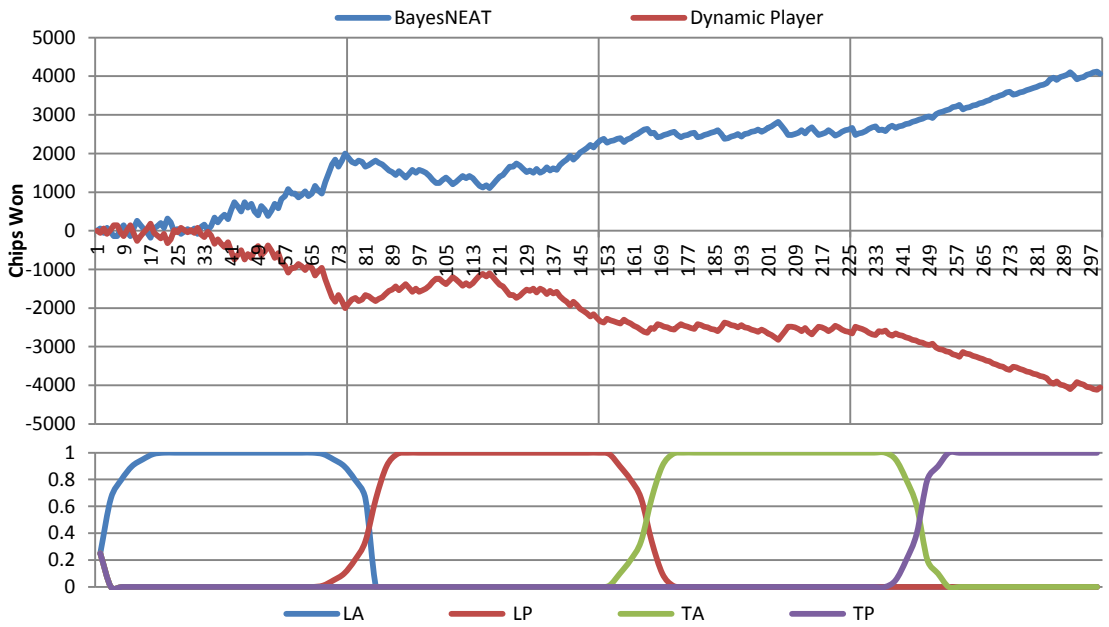


Figure 7.27: Performance of evolved agent against a dynamic opponent (with representation of player's belief of opponent style).

We can see in Figure 7.27 that our player is able to cope well against the style-changing opponent. It is notable that our player becomes susceptible to a change in strategy change as evidenced by the gradual decrease in winnings once the opponent style changes – Once the opponent model recovers (seen in the lower half of Figure 7.27) the agent's winnings also start to increase once again. The main reasoning behind this is that our Bayesian model has to readjust its style-representative weights in order to accommodate the opponent's shift in style, and as such a lag is involved in adapting to the opponents' strategy. This lag would be much more noticeable if we did not use the windowed observation covered in Section 0. This appears to show that changing from one style to another can be useful in deceiving an opponent until they become aware of the change in strategy. This approach is important in human play of Poker, in order to avoid predictable play, as well as exploiting the gullibility of the opponent (Sklansky, 1992). The failing of this approach against our ANN strategy, however is that the probabilistic way in which our modeler updates its beliefs means that these beliefs will be altered significantly when a tight player repeatedly performs an action that it should rarely do (the main example being to move from TP to LA, drastically increasing the frequency of betting actions; see Table 7.2: Action probabilities for each opponent style per betting round.).

7.9 Multi-Player Environment

Chapter 4, section 4.3 investigates the use of recurrent node connections in evolved ANNs to utilise data from multiple opponents. In our experiments will full-scale Poker

we make no exception, only the complexity of the problem has been increased. Research into Poker has mostly looked into two-player, one-on-one games (more commonly known as 'heads up' Poker) (Billings et al., 2003), (Findler, 1977), (Billings et al., 2002), (Barone & While, 1998), which reduces evaluation complexity, especially in relation to dealing with opponent models. The increased complexity of a 3 vs. 1 game, for example, calls for our neural network to interpret the information of 3 opponents instead of 1 (Sakaguchi & Sakai, 1992). We believe that the ability of NEAT to evolve network topologies as well as weights might lend itself to such a problem, particularly due to the strong possibility of creating *recurrent* neural networks. Our aim of reacting to more than one opponent shall take advantage of the 'memory' afforded to us through the recurrent connections (Elman, 1990), (Arvandi et al., 2008), (Bodén, 2001). We use the same network inputs as Figure 7.7, with the addition of a single input that represents the *position* of the opponent at the table (this is a binary input that is set to 0 if the opponent is yet to play after the agent or 1 if the opponent has already performed their action). We then iteratively pass the inputs for the first, second, and third opponent (in theory this could extend up to an n^{th} opponent). If evolution allows, this will result in a 'memory' of the previous opponents which should influence the current decision. After the final opponent's data is input, the output is received, and the action represented by the largest numerical output is performed (through analysis of the outputs we have seen that, after evolution, these values usually give a very clear decision). The fitness f for this experiment is represented by Equation (4), where a represents the number of player styles, C represents the number of chips won after

300 hands against player a, C_{MAX} represents the highest fitness of the a fitnesses, and C_{MIN} represents the lowest fitness of the a opponents played.

$$f = (\sum_a C)a^{-1} - (C_{MAX} - C_{MIN}) \quad (5)$$

This is designed such that our player is rewarded for playing well against all styles of opponent, rather than favouring some styles over others (the effects of which can be seen in Figure 7.28 and Figure 7.29). We evolve our agent's ANN against all four types of opponent (vs. 3 LA players, then 3 LP players and so on). In order to avoid the recurrency 'remembering' the order of opponents, we clear all data within the network between evaluations, and also force our agent to face all four styles in a random order. If we consider this evolution without our opponent model, we can see the results in Figure 7.28.

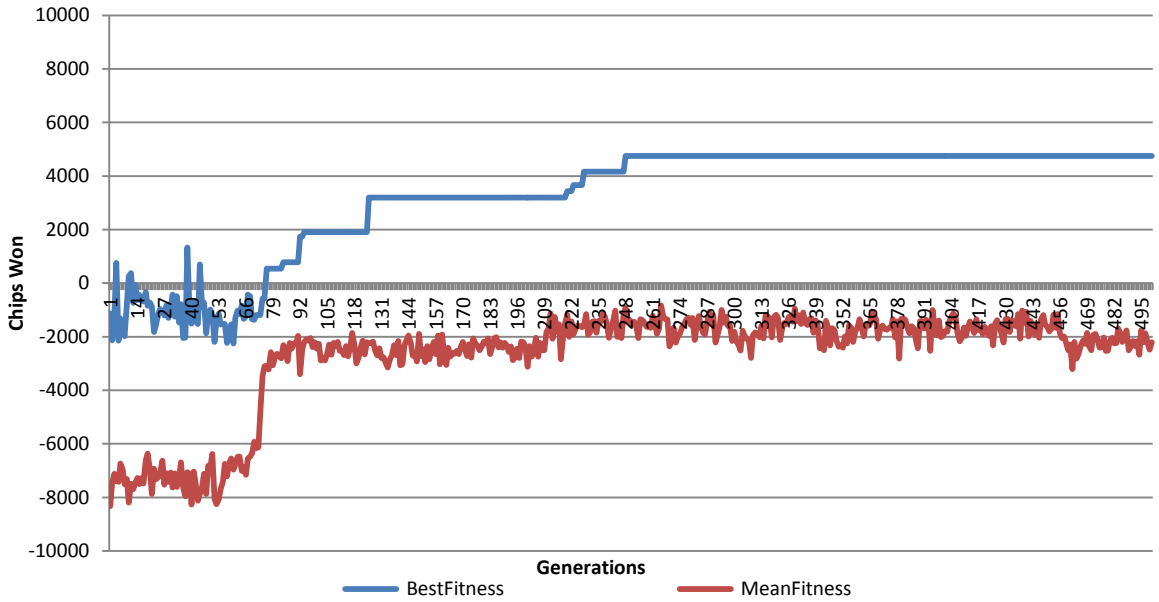


Figure 7.28: Evolution of a recurrent network (without opponent model) against all four types of opponent in a four-player scenario.

As can be observed, the player evolves to a successful level of chips won after 500 generations, but the number of chips won is comparable to that of Figure 7.20, which is against a *single player*. Considering each hand is versus three opponents in this instance, we would expect more sizeable winnings from a successful player. We now observe the evolution of a player using our multi-round Bayesian opponent model in Figure 7.29.

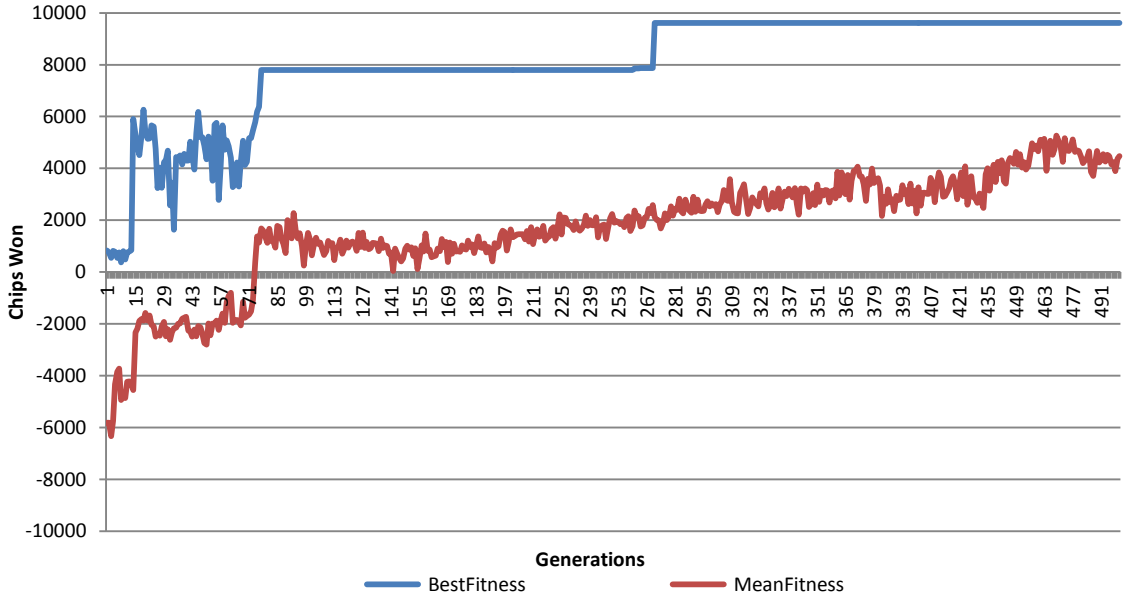


Figure 7.29: Evolution of a recurrent network (with opponent model) against all four types of opponent in a four-player scenario.

We notice that the quality of solutions found by NEAT's evolution using our opponent model have a drastic improvement over those shown in Figure 7.29, and even towards the 500-generation limit, appears to have the potential to evolve further as indicated by the average fitness. In the interest of ascertaining the quality of our recurrent ANN in comparison to an ANN using individual inputs per player, we evolve an agent that uses a similar design to Figure 7.7, but for each opponent (three in this case), we have eight inputs – three describing the last action of the opponent, four describing our

opponent model belief of the opponent (LA/LP/TA and TP respectively), and a single input denoting the position of our opponent yielding a total of 30 inputs. This network design can be seen in Figure 7.30.

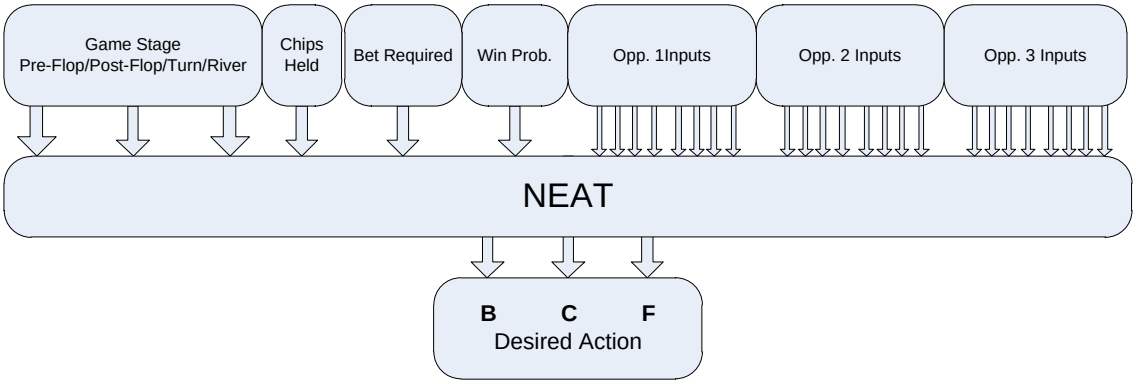


Figure 7.30: Design of a 30-input ANN.

We use the same fitness as Equation 4, and the results of evolution can be seen in Figure 7.31.

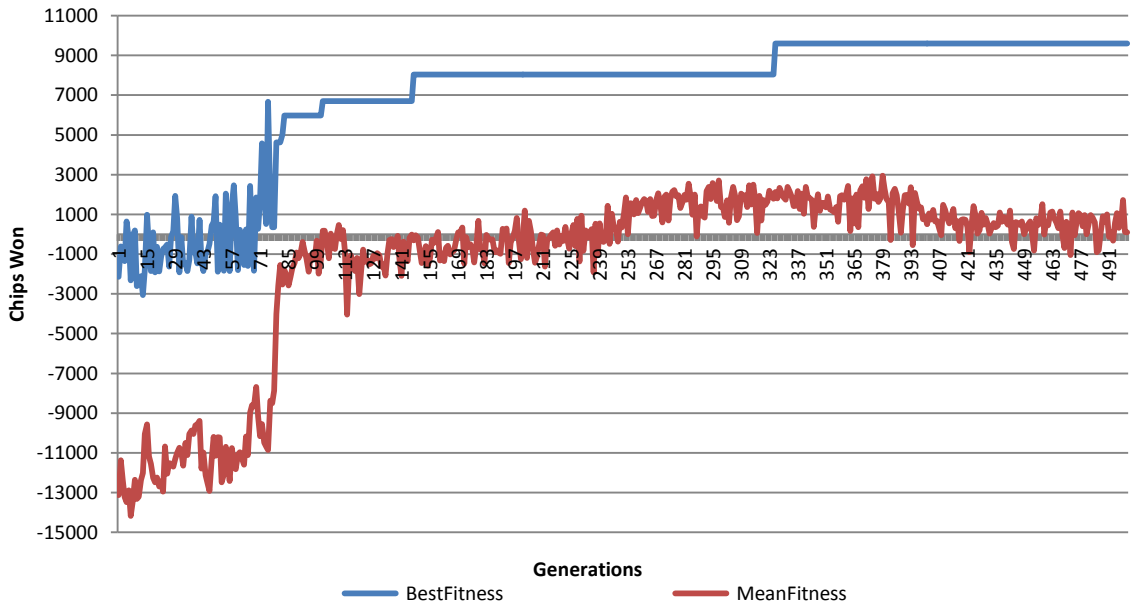


Figure 7.31: Evolution of a non- recurrent, 30-input network (with opponent model) against all four types of opponent in a four-player scenario.

We can see that the best fitness of our evolved agent using the 30 inputs is indeed comparable to the evolution of the recurrently evolved agent. The major difference between the two approaches is a *temporal* one such that the large number of inputs cause a slowdown in terms of the evolutionary process.

In Figure 7.32 we illustrate the tournament success of the best solution from the evolution illustrated in Figure 7.29, against all possible mixed player-table combinations, and as we can see, the network has excellent success rates against all possible opponent combinations including those it *was not trained against*. This is arguably due to the advantage a player receives once it has access to beliefs about an opponent's style of play. This advantage was shown in Section 5.3.1 using our simplified version of the game, where an ablative study (in Figure 5.24) showed that the removal of opponent model inputs have a seriously detrimental effect upon the performance of the agent. It may be safely assumed that the same conclusion in a similar study upon the agent used here (in an admittedly more complex environment) would also ring true. Using Bayesian probabilities in this environment appears to be vital to the success of the player, especially when aiding the generalisation of our player against unseen combinations of opponent.

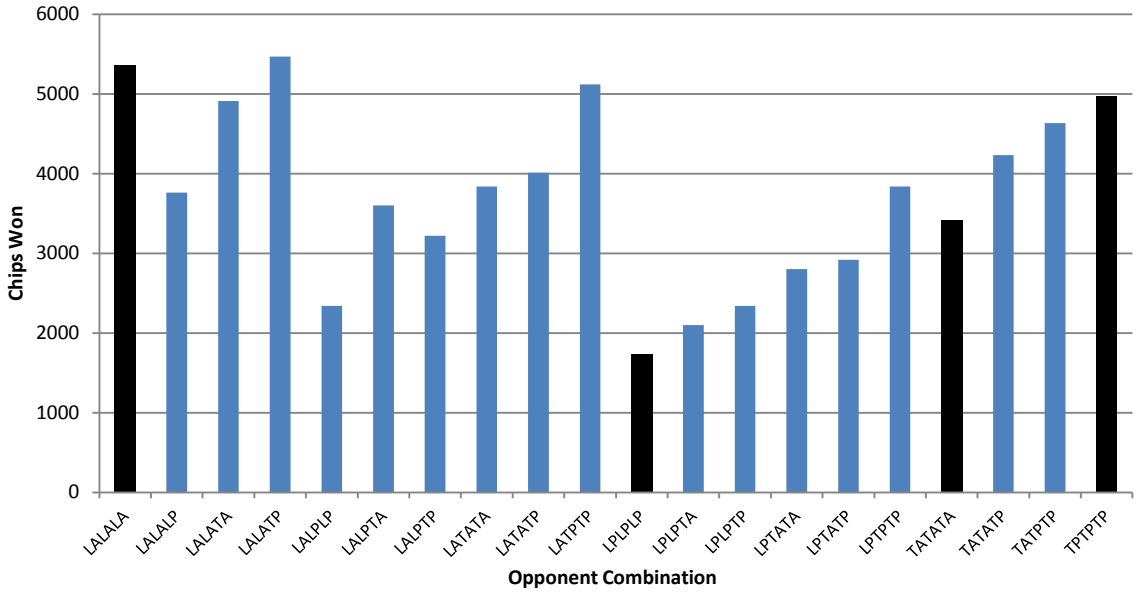


Figure 7.32: Tournament performance of the best evolved genome against all combinations of 3 opponents over 300 hands – The player was evolved against only the four combinations of players highlighted in black.

7.10 Discussion

The collection of action probability distributions is straightforward in a variety of game environments. Using a hard-coded or parameterized set of possible opponent strategies and Bayesian analysis we determine opponent style based upon previously observed actions. Bayes' theorem, which we have previously shown effective for a simple Poker game in Chapters 4, 5, and 6, has the potential for broad application in other games where discrete actions can be related to homogeneous strategies (or goals as demonstrate in Chapter 3).

After adaptation to a multi-round environment, we find that the update of opponent beliefs is a clear progression given a set of style-specific actions (demonstrated in Figure 7.16 through Figure 7.19) yielding a sensible change in strategies as the game

situation changes (as long as the hard-coded strategies are clearly defined and broad in their scope), as well as being relevant to the player's actions.

This approach assumes that we have a collection of known strategies to compare against, but as shown in this (and previous) chapters, our models need only represent a spectrum of possible opponent models (which need not include the actual player strategy) that do not necessarily have to be entirely correct or complete to be useful.

7.11 Summary

In this chapter, players are evolved using NEAT for Texas Hold'em Poker. We first show it is easy to evolve a player against individual opponents of a fixed style. We then compare the improvement in performance of our agents afforded by the use of opponent models in aiding the evolution of game-playing agents against players that do not make use of such information. Against a single adversary, the results show little difference between approaches that use an opponent model, and those which do not. We take into account the multi-round basis of full-scale Texas Hold'em and utilise a round-based Bayesian model which adjusts beliefs based upon opponent behaviours given the specific round and action style probabilistic pairs for that round.

We compare our players in an environment where there is more than one opponent. Results show the benefits of using opponent models increase with a greater number of opponents. In this instance, the evolved player relies upon Bayesian opponent modeling and the recurrent nature of the evolved neural network is shown to be

crucial in order to apply the appropriate strategy for each set of different opponents, and is able to generalise and defeat tables of opponent combinations not yet encountered. Furthermore, we test the approach against opponents that employ simple bluffing tactics, as well as simple dynamic strategy approaches. In these experiments we find that our opponent-model augmented NEAT networks are able to perform well against these dynamic opponents.

Chapter 8: Summary and Conclusions

This thesis presents a Bayesian approach to opponent modeling in a series of games with both perfect and imperfect information, as well as investigations involving the robustness of the given model using Coevolutionary and reinforcement approaches. This work exemplifies the useful nature of opponent models in various game playing environments, and as such shows that the understanding of an opponent's behaviour, even if the *actual* behaviour does not necessarily adhere to the inferred behaviour, can prove useful for the agent making use of the model.

The application of Bayesian analysis to various game environments is a generalizable approach, as long as the game environment is amenable to observation, such that intentions can be gleaned through the use of Action|Goal or Action|Strategy beliefs and the conditional independence for each of the Goals or Strategies – the range of problems these beliefs can be applied to is wide. Examples can range from that such as a game of football, where beliefs of opponent strategy relating player movement and overall tactical intention can be inferred. Contrast this to a First Person Shooter environment where short term goals may be assessed through a small set of actions – if we consider the A* pathfinding approach used in Chapter 3, we can consider that this may be upscaled to deduce opponent intentions such as 'pick up stronger weapon' or 'pick up health pack' and intercept (as in Chapter 3's experimentation) or apply a combative strategy to ruin the opponent's plans. The understanding of a potential

opponent goal yields a potentially beneficial avenue for a counter strategy, and the application over various game environments is discussed further in Chapter 3, Section 3.5.

The Action|Strategy beliefs employed in Chapter 3 are developed through the use of terrain analysis, which can be a useful tool in real-time perfect information strategy game environments – the coordination of Bayesian analysis with the directional behaviour of an opponent's potential path proves a useful tool for goal analysis given a small set of discrete actions. A further benefit of using the A* algorithm to generate Action|Strategy data is that the collection of probabilities will always be conditionally independent as long as (disparate) potential goals do not occupy the exact state space, therefore the collection of directional moves in comparison of one goal to one another remains heterogeneous. The robust nature of the Bayesian approach (in regards to the Halmoids environment) is tested through a randomized opponent that randomly performs actions outside of the predicted set (Section 3.4, Figure 3.6), which reduces the analysis accuracy by only a small amount, with the analysis agent succeeding in its interception to a significant frequency.

In Chapter 4 we introduce a simple version of the game of Poker, consisting of a 10-card deck, and one card per player with a single betting round and a 'highest card wins' win scenario. We develop and describe four interdependent opponent styles as well as determining the means to defeat these styles to a sufficient degree. Section 4.2.4 introduces the application of our opponent model into an imperfect information

scenario, using an observationally-calculated set of Action|Style probabilities in conjunction with Bayes's rule and opponent actions in order to determine a suitable counter-strategy. Facing a single strategy can prove simple to beat, but a collection of strategies at the same table represents a more complex problem that is handled using a system of opponent strength ranking and response to the most commonly used style at the table. The result is understandably imperfect, but copes sufficiently (and better in some cases) than an opponent which is privy to the *actual* style of all opponents, rather than just the Bayesian analysis of assumed opponent type.

To combat the weakness of compromise introduced with the Pigeonholing approach, as well as the limitation of using a simple two-parameter strategy to combat the behaviour of another two-parameter opponent, an evolutionary approach to player decision is applied in Chapter 5, Section 5.1. Initially, the design of an agent using a chromosome evolved by Genetic Algorithm as its decision function is developed, and in attempting to evolve a suitable player against all opponent types, a disappointingly weak overall player is evolved. The use of our Bayesian opponent model is applied to the solution (resulting in a significantly larger chromosome), but the resulting tactic selection due to Bayes' rule tenders a greater performance in comparison to an evolutionary approach which does not make use of an assumed opponent strategy through modeling.

The size of the chromosome is a limiting factor for generalization over multiple opponents, as well as situations where more than the given set of styles is employed

(for each potential style, another set of genes to determine counter strategy needs to be added). This limitation of size in respect to generalizability is considered through the introduction of a Neuroevolutionary approach in Section 5.2, where the application of the opponent model once again yields a stronger multi-style player with respect to a neuroevolutionary agent in ignorance of an opponent model. This strength is further expounded through an investigation into the use of the recurrence present within many evolved ANNS to recognize and respond to multiple agents using a set of inputs that accommodate for only one opponent. The sequential passing of opponent data results in a successful player in a multiplayer environment regardless of the small number of inputs (thus negating the explosion of size per opponent associated with an ever-expanding chromosome), such that in the analysis of the resulting strongest player, the model aids strong performance against collections of players that it has not been evolved against.

The testing involved in Section 5.4 (as well as Chapter 6) correspondingly tests this claim, applying bluffing tactics to the static players in an attempt to deceive the modeling process, as well as test the strength of the agent and model given a non-style-specific opponent. Chapter 6 expands the testing of the robustness of our Bayesian model by using a coevolutionary approach to developing a collection of agents which are strong (in relation to one another), but not of a distinct style. The 'best' evolved Bayesian ANN player from Chapter 5 is tested against each member of both populations to determine how the agent responds to numerous unknown

opponent types, as well as determine the style with which it believes each opponent to be. The Bayesian ANN agent is shown to respond well to the disparate collection of opponents, of which some also perform well against the initial static styles (which our ANN has been trained against).

Chapter 7 expands the investigation of the effect of an opponent model upon evolution in the expansive realm of full scale Texas Hold'em Poker. The application of a model intended for a single round over the multi-round tournament initially delivers evolved agents that appear to perform successfully, however through analysis of play behaviour it is seen that a decline of success is exhibited. We adapt our analysis over the separate rounds to understand that as play progresses win probabilities are greatly altered based upon the style of play. An example is that of a Tight Passive player which will mostly only aim to see later betting rounds if a good hand is held and will therefore fold less frequently in later betting rounds than in those earlier. This adaptation improves the accuracy of the model, and subsequently the same conclusion is reached that although the (single-round) Bayesian player performs better than a non-modeling player, a model must be applicable to several separate instances of play. The separation of individual sets player behaviours is shown important over separate betting rounds, as the tactics of a players strategy will adapt given the improvement (or decline) of a players hand given new community cards. Once again the effectiveness of recurrence in the evolved ANNs is used to generate solutions which

are effectual in both multi-player environments and against players which adopt varying levels of bluffing frequency.

8.1 Future Work

It is recommended that the following domains would be interesting areas within which to expand this research in the future:

8.1.1 A Preference-Based Modeling Approach to Real-Time Game Environments

The work presented in this thesis (in Chapter 3) exemplifies the use of Bayesian opponent modeling as a successful technique to predict low-level goals in relation to a limited set of basic movements. We suggest that this approach can be expanded in several directions to judge other facets of overall plans or potential sequences within an opponent's play. An example of this (when considering an interactive game environment) could involve the analysis of potential goals of varying function or importance. Whereas the work in Chapter 3 considers that all potential goals are of equal importance to the opponent (such that no one goal is advantageous over the other) this work could concentrate upon a perceived *preference* of goal as well as predicting a potential order of goal visitation. For example, consider a multi-player First Person Shooter (FPS) environment such as UNREAL TOURNAMENT, where local objectives (such as finding a more powerful weapon, and restoring lost health) may be more immediately tended to in comparison to an overall goal such as either killing an

opponent or capturing an enemy base. A ‘preference of visitation’ could be considered based upon the status of an opponent, such as having started the match where the collection of a stronger weapon, or advancing towards an enemy goal would take priority, or in relation to nearing the capture of an enemy base where restoring health and defeating opponents would be the primary target. The prediction capability added to the QuakeBot, for example, used its own decision processes to determine what it would do if in the same situation as the opponent (Laird, 2001). In this case however, the prediction of how to respond to an opponent’s targets can utilize the Bayesian analysis of opponent behaviour, with an extension into an analysis of the preferential ordering of the opponent’s goals. The analysis could use the low level behaviour (such as movement direction with respect to a probabilistic distribution) to determine the low level goal of an opponent (as with the work in Chapter 3), but also take the completion of low level goals as an indication of the intended higher level tactical goal. The work would be best served through an investigation of the work of Sander Bakkes and Sushil Louis particularly where the game scenario investigated involves overarching strategic goals, and low level goals are completed in order to achieve those higher goals (Bakkes et al., 2004) (Bakkes et al., 2005) (Louis & Miles, 2005) (Miles & Louis, 2005).

8.1.2 A Bluff-Aware Approach for Modeling an Opponent's Potential for Deceptive Play

Chapters 5 through 7 indicate the strength of our model's stability in the face of unseen collections of opponents, as well as those which enact bluffing behaviours with a given probability, *p*. We have also performed brief investigation into how a set of behaviours can be input to an evolved Artificial Neural Network in order to determine the opponent's play style. Our model is indicative of the opponent's play style using only the previous actions performed by the opponent, but the *outcome* of these performed actions could also be used in order to make a model of how the opponent performs his deceptive strategies (such as bluffing). The aim of this further investigation could investigate the effect that using this information has upon the stability (and respective success) of an agent, evolved or otherwise. For example, if we take into account the use of Blind Stealing, where an opponent with a weak or moderate strength hand decides to perform an aggressive action in order to force opponents out of the hand. If this approach is performed by a primarily tight player, the likelihood that the other players fold is increased. With this investigation, situations where a bluff is called, or a blind stealing hand is revealed, or even where an opponent may win the hand but quite obviously has had a weak set of hole cards up until the final betting round, could be recorded and stored as a frequency/probability of performing a deceptive tactic per hand. Situational data could be taken into account, such as the size of the player's purse in relation to the deceptive strategy, or

the outcome of a previous hand. Research has shown the predictability of humans in relation to behavioural patterns is high and therefore reactionary behaviours such as going 'all-in' after a hand with a heavy loss in order to try and recoup some chips may be indicative of an opponent's mental makeup (Song et al., 2010). The methodology may involve the use of the Bayesian ANN approach explored in this thesis, but could further analyse the effects that incidental and deceptive outcome data (as inputs to the network) have upon the responsive nature of the network to adapt to a deceitful opponent's play. A limitation of this observation may ironically be due to the use of deception; if an opponent uses plays such as the 'blind stealing' approach successfully, the act of 'mucking' (hiding the hole cards when a showdown is not reached) would reduce the ability of recognizing when an opponent has actually bluffed in order to win the hand. It is recommended that the act of mucking be ignored during initial investigation in order to see the effectiveness of such frequency data when all cards are made visible at the end of a hand. The expansion into hidden card data may also reveal interesting information as to the nature of how often bluffing plays can be spotted without always having access to information about the bluffer's cards (as well as success).

8.1.3 Look-Ahead Prediction of Potential Strategy Adaptation

The work presented in this thesis primarily concerns the determination of an opponent's strategy based upon the collection of actions *already performed* by said

opponent. This is in contrast to Davidson's approach, which uses information about current action frequencies in order to determine the opponent's *next* action (Davidson et al., 2000). This being said, given Davidson's claimed accuracy of prediction (over 80% accuracy), an interesting avenue of research could be to use a predictive capability to determine an opponent's next action, and use this action as part of the belief model of an opponent's *current* style. If we consider our Bayesian approach, the set of previous actions used determine the current style, but a predictive capability may infer a potential 'future style'. The proposed investigation would involve an analysis of *how far* forward into the hand a predictive ANN could look as to the opponent's potential actions; this could then be used as a potential early indicator of a switch in opponent style. This approach could also intertwine with that from 8.1.2, where a predictive capability could infer whether the opponent was preparing to bluff, such as in the latter stages of a hand where an enterprising player may imply that the river card was a necessary card to make a strong hand available to him.

8.1.4 Modeling Strategy Recognition Through Idiosyncratic Behaviour Capture

As previously mentioned in Section 8.1.2, human behaviour can be predictable, and the dynamic and interesting nature of the human (face to face) game of Poker may be attributed to the ability of a player to see more than just the opponent's previous actions and betting behaviour. A good human player can also look for 'tells' (seen as a behavioural tic in relation to a given situation) that a human opponent may display,

subsequently deciding upon the legitimacy of the opponent's actions. This analysis of behaviour is far less common in relation to computer-based (or online) Poker, and as such the question of how idiosyncratic behaviours of a human whilst using the *peripherals* of a computer may indicate the human player's hand strength. Some humans display compulsive behaviours to some degree, which can be exacerbated by stressful situations, research has shown that Poker players have been known to use medication such as beta blockers in order to calm nervous behaviour as well as improve concentration (Nova Southeastern University, 2010). This investigation may consider capturing a player's computer mouse movement (or clicks) during hands, and through the analysis of hand outcomes with respect to input behaviours determine a player's hand strength, or recognize situations where a bluffing tactic may be employed. A collection of the frequency of potential behaviours performed could be collected, for example the frequency of mouse clicks and certain abnormal mouse movements (such as shaking). These frequencies may be related to a post-hand analysis of outcome given the biometric collection of player data (i.e. if a player clicks compulsively, and wins the hand, this behaviour could be inferred as a sign of a good hand). Limited research has been performed in the area of biometric capture with respect to game playing, generally looking only to identify users through behaviour patterns (Kaminsky et al., 2008), but the inference of opponent style and tactic through (partially unintended) behaviours may prove interesting.

Glossary

A*: An algorithm widely used in pathfinding and graph traversal, uses heuristics to calculate the shortest path from one node to another

A Priori: Prior knowledge, commonly used by Bayesian analysis to make inferences conditional upon this knowledge

AI: Artificial Intelligence: The study and design of intelligent agents.

ANN: Artificial Neural Network

Ante: A forced bet from all players in the game of Poker before a hand can begin

Blind: A forced bet from one or more players before a hand can begin

Bluff: A misleading betting action in a game of Poker.

Boolean: A value which can take the value of *true* or *false* (1 or 0)

Deathmatch: A common multiplayer videogame mode where two or more players kill one another's avatars to score points.

EA: Evolutionary Algorithm

Flop: Three initial community cards given after an initial round of betting in a game of Texas Hold 'em

FPS: First Person Shooter, An action game in which the player controls the protagonist from a first person viewpoint.

GA: Genetic Algorithm.

Heads Up: A Poker game which consists of only two players

Heuristic: An experience-based technique used for problem solving, a 'rule of thumb'.'

IPD: Iterated Prisoner's Dilemma

Nash Equilibrium: In a game of two or more players, a Nash Equilibrium is reached when each player knows one another's strategy, and no player can gain from changing from their current strategy.

NEAT: NeuroEvolution of Augmenting Topologies

NeuroEvolution: A form of machine learning that uses evolutionary algorithms to train neural networks

River: A fifth community card in the game of Texas hold 'em

Robustness: The ability of a computer system to operate despite abnormalities in the input.

rtNEAT: Real-Time NEAT. An implementation of the NEAT algorithm for a real-time game environment

RTS: Real Time Strategy. A game genre which requires the player to command multiple units to complete various objectives or defeat a given opponent (usually in a military scenario).

SharpNEAT: A C# implementation of NEAT.

Splash Damage: An area of influence from explosions in deathmatch games within which damage can still be inflicted upon a desired opponent.

Stochastic: Random, non-deterministic.

TIELT: The Testbed for Integrating and Evaluating Learning Techniques

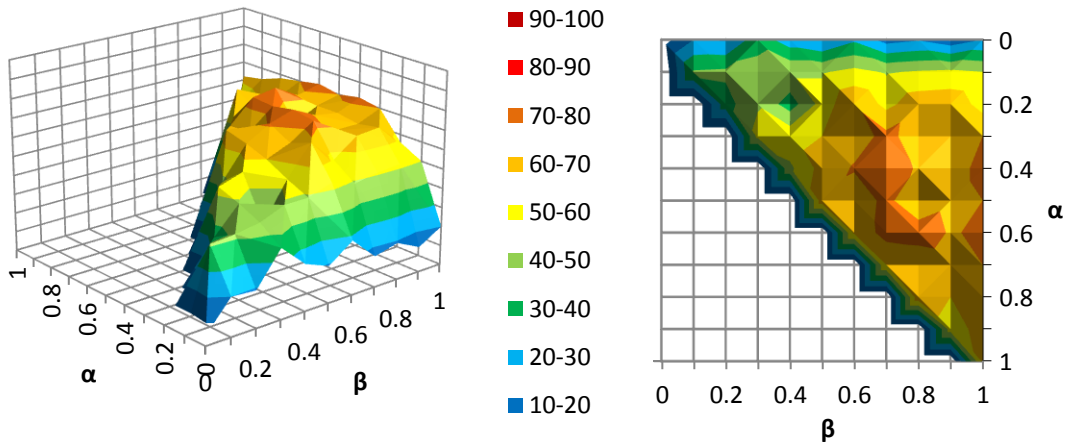
Tuple: An ordered list of Elements

Turn: The fourth community card in a game of Texas Hold 'em.

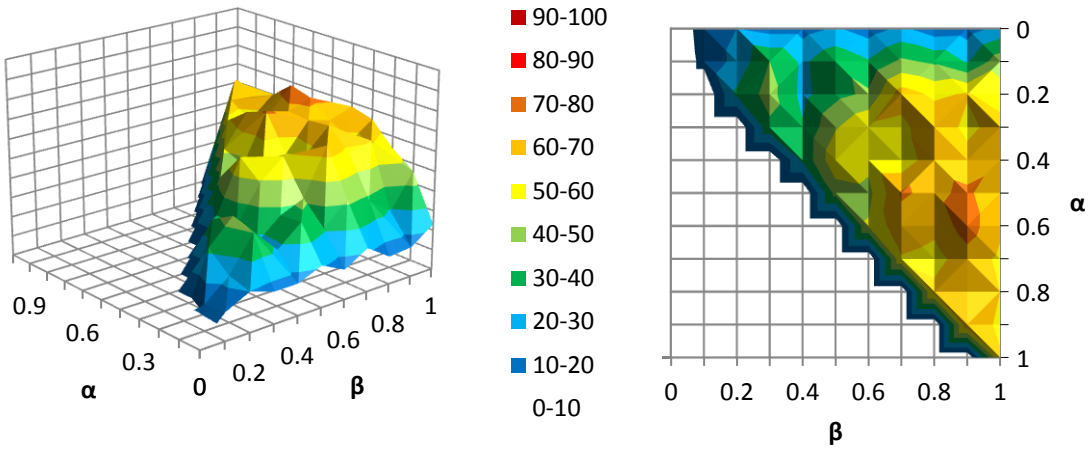
Appendix A

Performance of varying α and β values against a collection of varied opponent styles

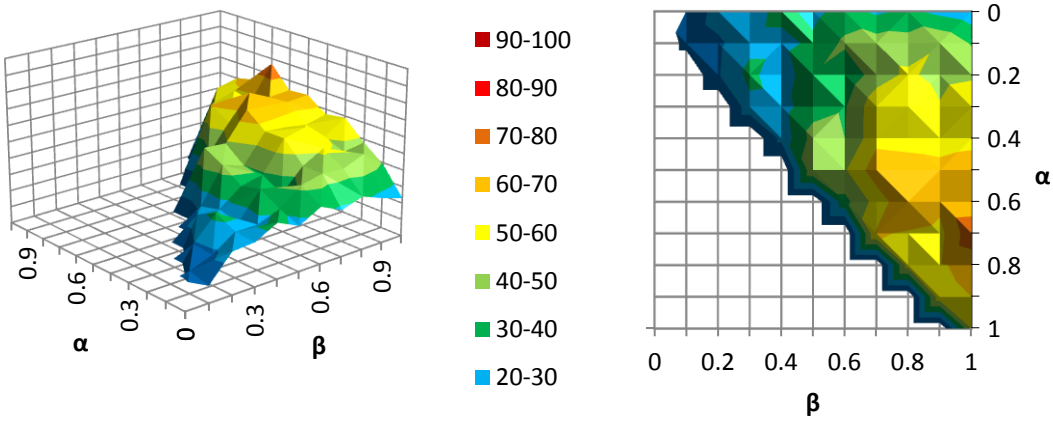
The following graphs are the result of 100 tournaments played between an 'amorphous' player, which accepts α and β values as an argument, and every combination of three opponents given the four potential opponent styles. The graphs show the performance of the amorphous player where the α and β values are individually incremented in steps of 0.1 from 0.0 to 1.0.



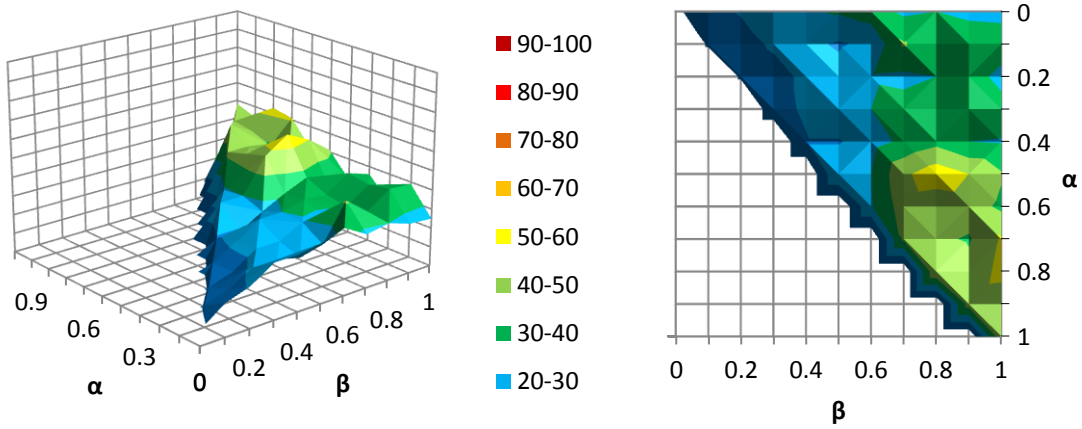
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LA/LA



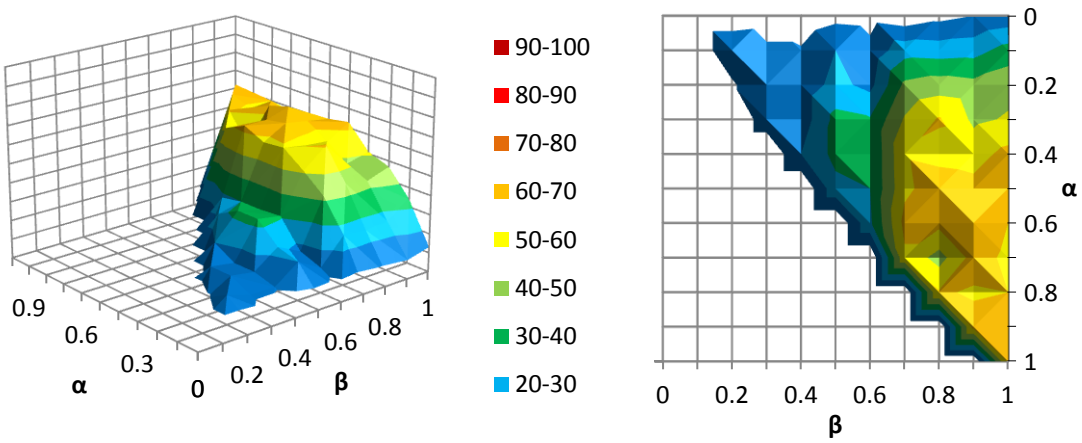
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LA/LP



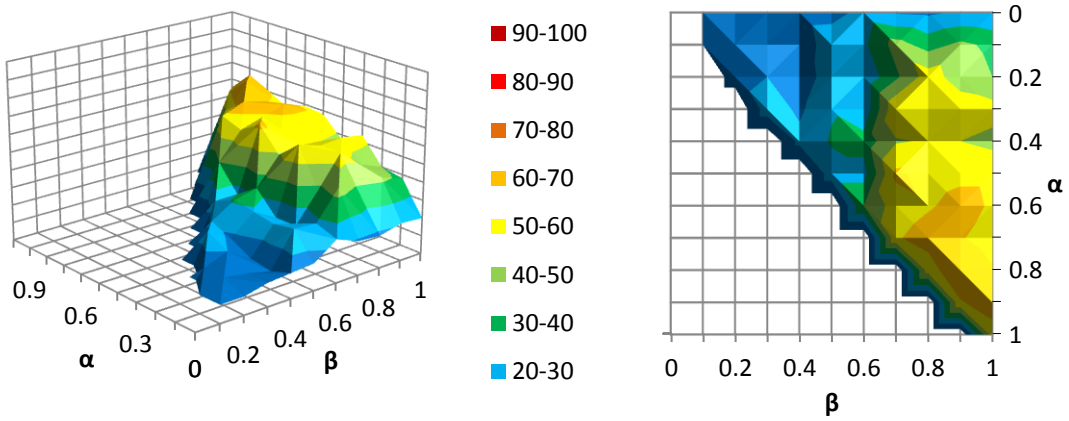
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LA/TA



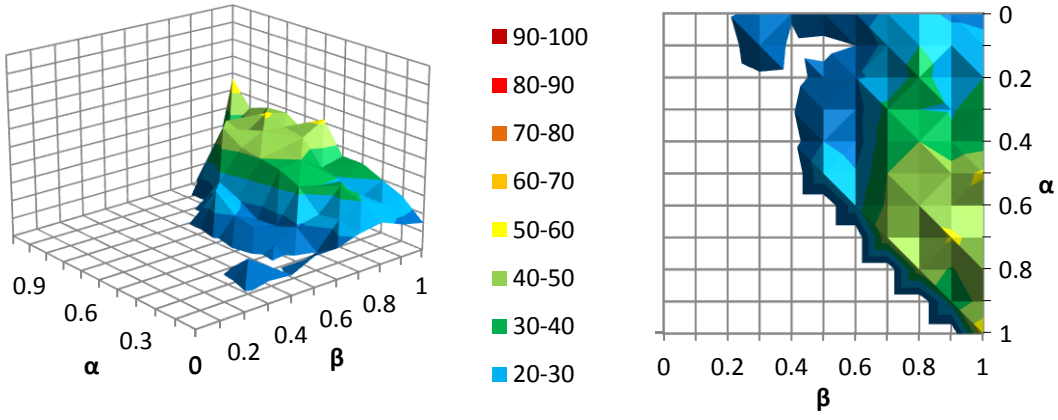
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LA/TP



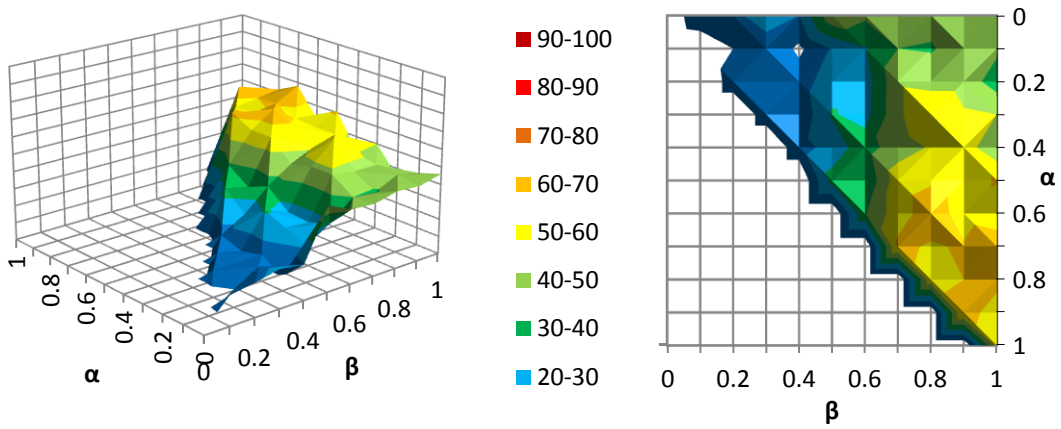
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LP/LP



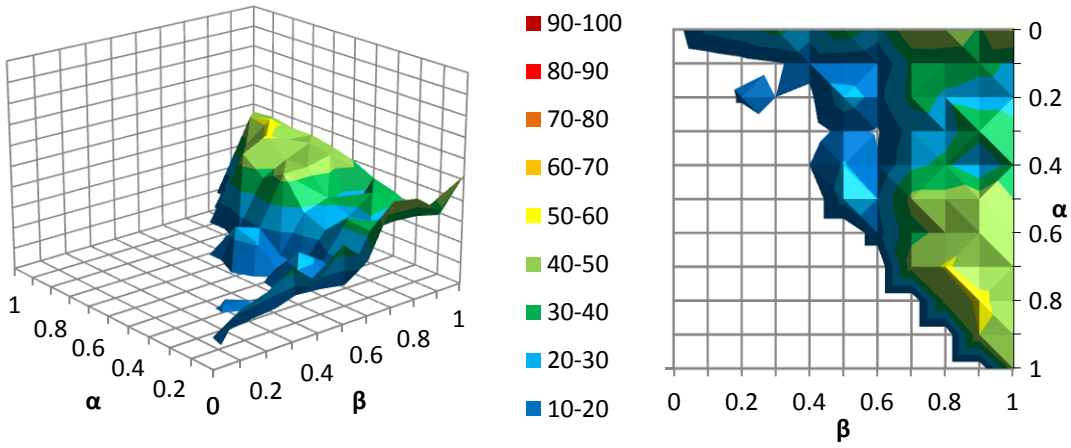
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LP/TA



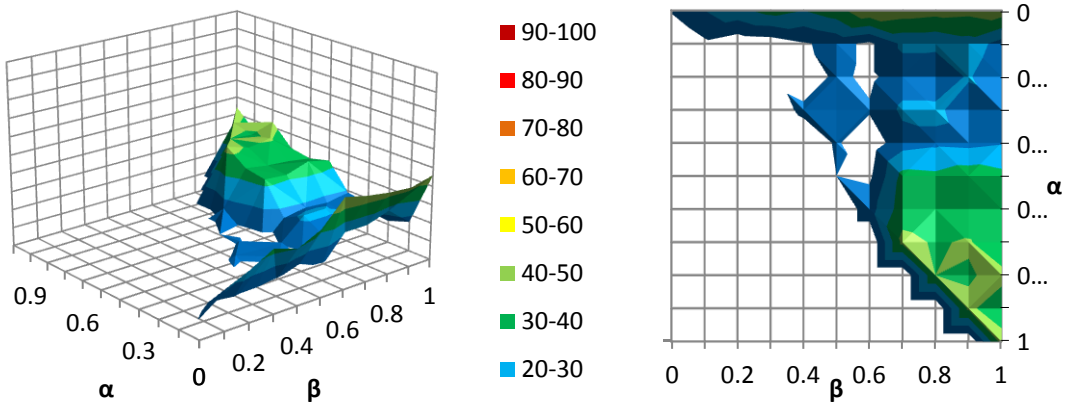
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/LP/TP



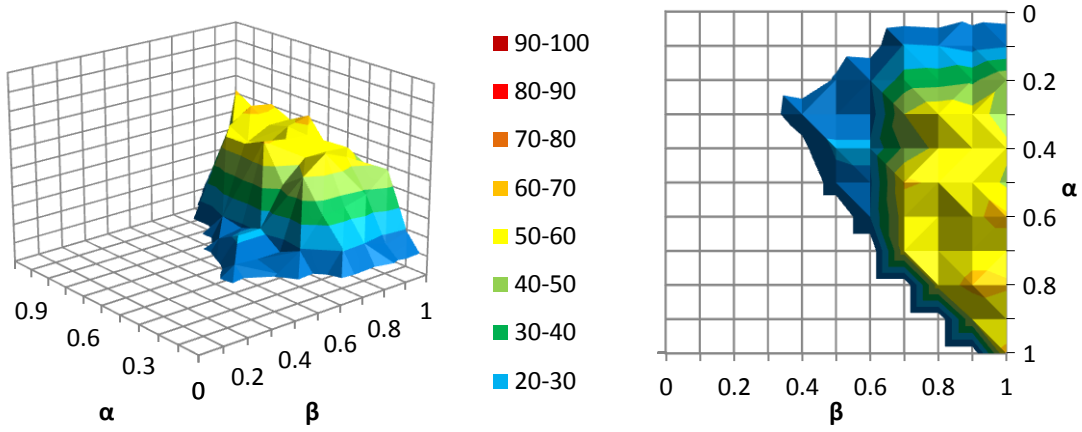
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/TA/TA



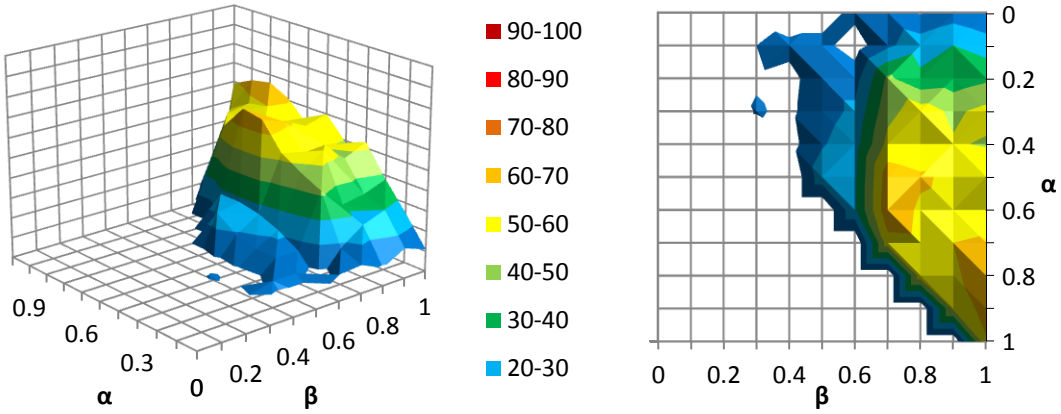
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/TA/TP



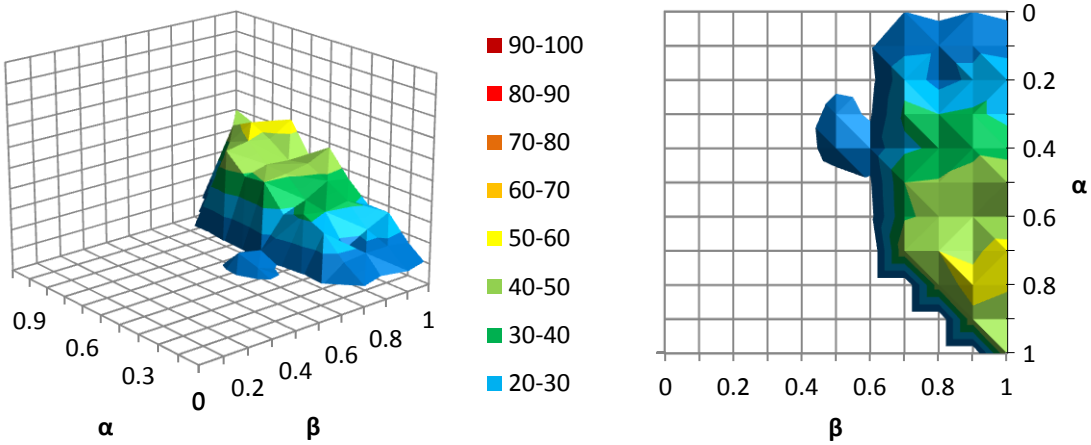
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LA/TP/TP



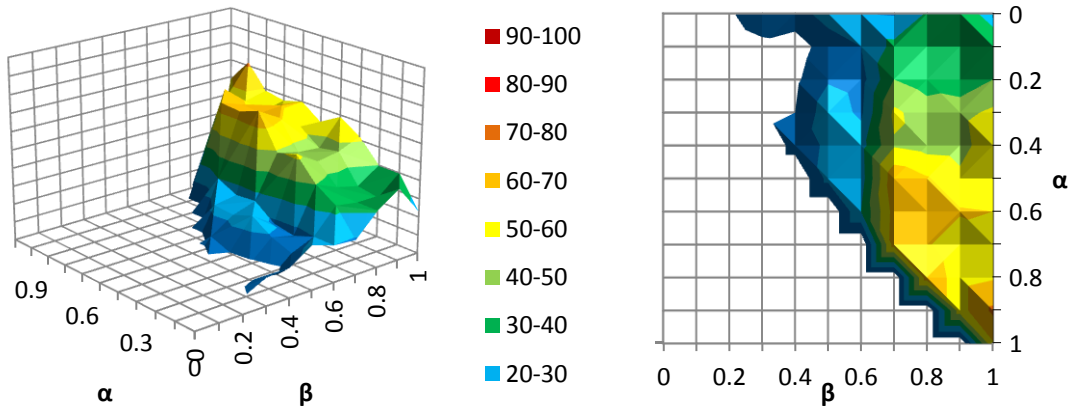
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LP/LP/LP



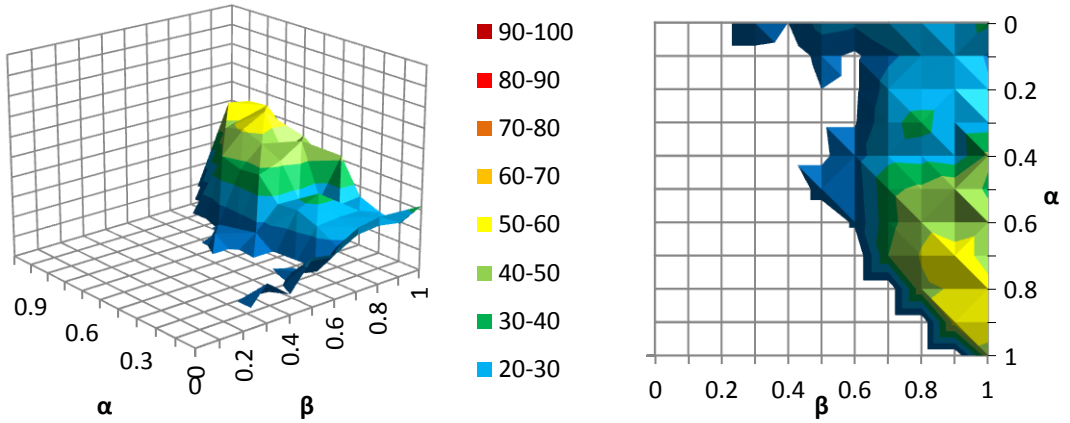
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LP/LP/TA



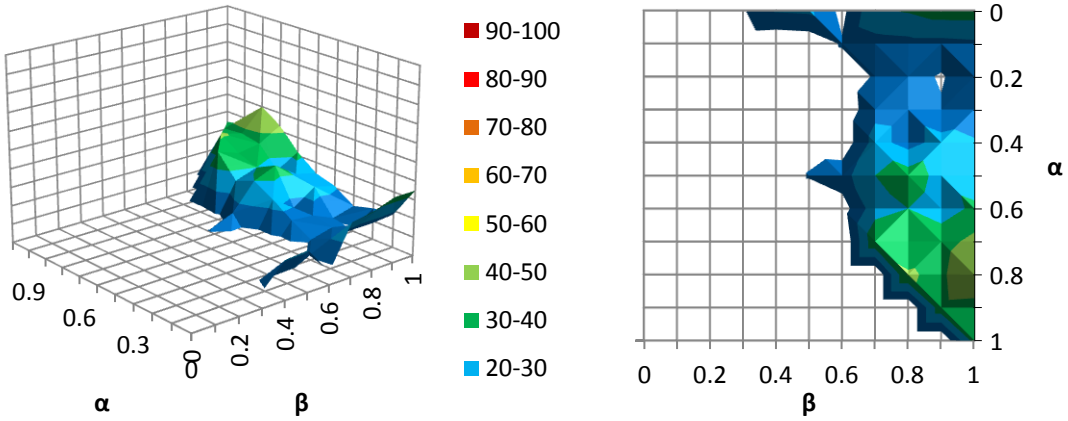
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LP/LP/TP



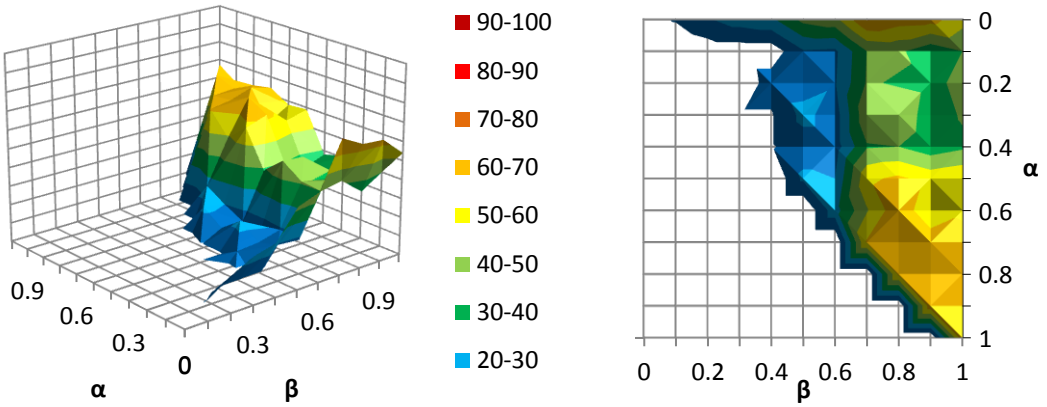
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LP/TA/TA



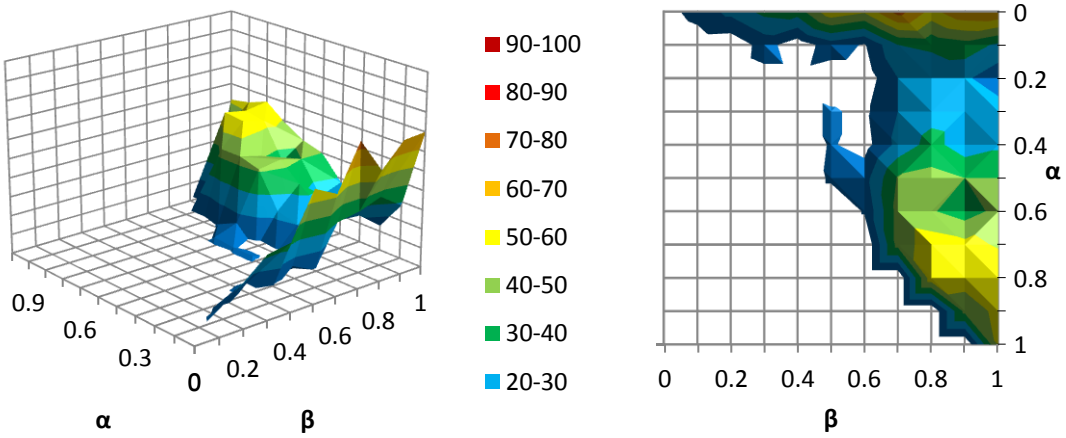
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LP/TA/TP



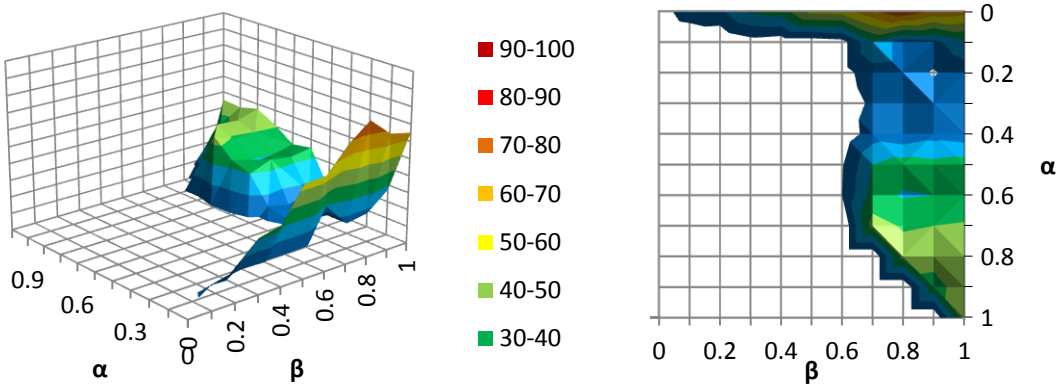
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against LP/TP/TP



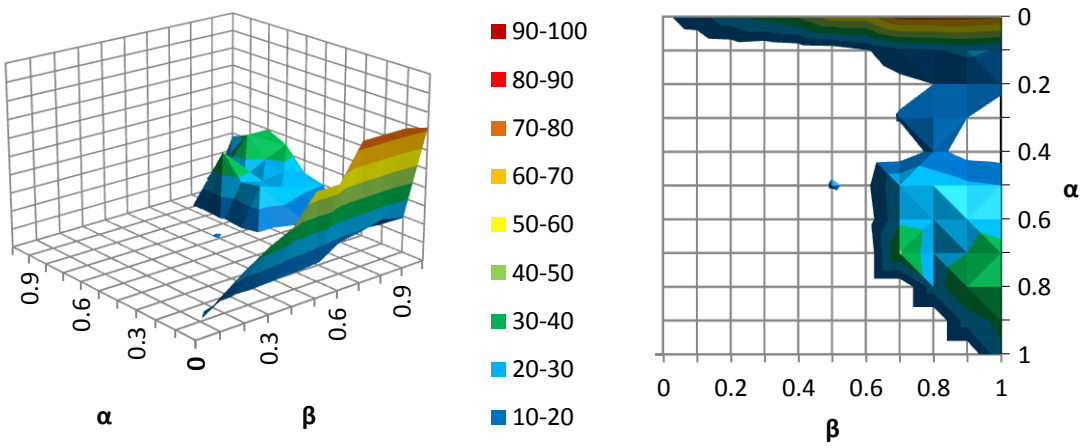
Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against TA/TA/TA



Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against TA/TA/TP



Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against TA/TP/TP



Tournament success of different values of α and β (in % of tournaments won) when playing a four-player tournament against TP/TP/TP

Bibliography

Alexander, T., 2002. GoCap: Game Observation Capture. In S. Rabin, ed. *AI Game Programming Wisdom*. Hingham, MA: Charles River Media. pp.579-85.

Arvandi, M., Wu, S. & Sadeghian, A., 2008. On the Use of Recurrent Neural Networks to Design Symmetric Ciphers. *Computational Intelligence Magazine*, May, IEEE Press. pp.42-53.

Aumann, R., 1959. Acceptable points in general cooperative n-person games. *Contributions to the Theory 23 of Games IV, Annals of Mathematics Study*, 40, pp.287-324.

Axelrod, R., 1987. The evolution of strategies in the iterated prisoner's dilemma. In L. Davis, ed. *Genetic Algorithms and Simulated Annealing*. Los Altos, CA: Morgan Kauffman. pp.32-41.

Baker, R.J.S. & Cowling, P.I., 2007. Bayesian Opponent Modeling in a Simple Poker Environment. In Kendall, G. & Lucas, S., eds. *IEEE Symposium on Computational Intelligence and Games (CIG 2007)*. Honolulu, USA, 2007. IEEE Press pp.125-31.

Baker, R.J.S., Cowling, P.I., Randall, T.W.G. & Jiang, P., 2008. Can Opponent Models Aid Poker Player Evolution? In Hingston, P. & Barone, L., eds. *IEEE Symposium on Computational Intelligence and Games 2008 (CIG 2008) 15 - 18 December*. Perth, WA, 2008. IEEE Press pp.22-30.

Baker, R.J.S., Cowling, P.I., Randall, T.W.G. & Jiang, P., 2008. Using Bayes' Theorem for Path Prediction. In Rigas, D., ed. *9th Informatics Research Workshop for Research Students, University of Bradford, 2008.* Bradford, 2008. University of Bradford pp.101-04.

Bakkes, S. & Spronck, P., 2005. Symbiotic Learning in Commercial Computer Games. In Mehdi, Q., Gough, N. & Natkin, S., eds. *CGAMES 2005, 7th International Conference on Computer Games*. Wolverhampton, UK., 2005. University of Wolverhampton pp.116-20.

Bakkes, S. & Spronck, P., 2006. Gathering and Utilising Domain Knowledge in Commercial Computer Games. In Schobbens, P.-Y., Vanhoof, W. & Schwanen, G., eds. *Proceedings of the 18th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC 2006)*. Namur, Belgium, 2006. University of Namur pp.35-42.

Bakkes, S., Spronck, P. & Postma, E., 2004. TEAM: The Team-oriented Evolutionary Adaptability Mechanism. In Rauterberg, M., ed. *Entertainment Computing - ICEC 2004*. Eindhoven, 2004. Springer-Verlag pp.273-82.

Bakkes, S., Spronck, P. & Postma, E., 2005. Best-Response Learning of Team Behaviour in Quake III. In Aha, D.W., Muñoz-Avila, H. & van Lent, M., eds. *IJCAI-05 Workshop on Reasoning, Representation, and Learning in Computer Games*. Washington, DC, 2005. Naval Research Laboratory pp.13-18.

Barone, L. & While, L., 1998. Evolving Adaptive Play for Simplified Poker. In *n proceedings of IEEE International Conference on Computational Intelligence (ICEC-98) April 6-9*. Seoul, Korea, 1998. IEEE Press pp.108-13.

Barone, L. & While, L., 1999. An Adaptive Learning Model for Simplified Poker Using Evolutionary Algorithms. In Angeline, P.J., ed. *In proceedings of the Congress of Evolutionary Computation (CEC'1999) July 6-9*. Washington DC, 1999. IEEE Press pp.153-60.

Barone, L. & While, L., 2000. Adaptive Learning for Poker. In Whitley, L.D. et al., eds. *In proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2000)*. Las Vegas, Nevada, 2000. Morgan Kaufmann pp.566-73.

BBC News, 2006. *Chess Champion Loses To Computer*. [Online] Available at: http://news.bbc.co.uk/1/hi/world/europe/6212076.stm [Accessed October 2008].

Billings, D., 1999. *The First International RoShamBo Programming Competition*. [Online] Available at: http://webdocs.cs.ualberta.ca/~darse/rsb-results1.html [Accessed January 2009].

Billings, D. et al., 2003. Approximating game-theoretic optimal strategies for full-scale poker. In Gottlob, G. & Walsh, T., eds. *18th International Joint Conference on Artificial Intelligence*. Acapulco, Mexico, 2003. Morgan Kaufmann pp.661-68.

Billings, D., Davidson, A., Schaeffer, J. & Szafron, D., 2000. Improved Opponent Modeling in Poker. In *Proceedings of The 2000 International Conference on Artificial Intelligence (ICAI'2000)*., 2000. pp.1467-73.

Billings, D., Davidson, A., Schaeffer, J. & Szafron, D., 2002. The Challenge of Poker. *Artificial Intelligence Journal*, 134(1-2), pp.201-40.

Billings, D., Papp, D., Schaeffer, J. & Szafron, D., 1998. Opponent Modeling in Poker. In *In Proceedings of AAAI-98 (15th National Conference of the American Association for Artificial Intelligence (AAAI))*. Madison, WI, 1998. AAAI Press pp.493-99.

Billings, D., Peña, L., Schaeffer, J. & Szafron, D., 1999. Using probabilistic knowledge and simulation to play poker. In *In Proceedings of the Sixteenth National Conference on Artificial intelligence and the Eleventh innovative Applications of Artificial intelligence Conference innovative Applications of Artificial intelligence*., 1999. pp.697-703.

Billings, D., Peña, L., Schaeffer, J. & Szafron, D., 2001. Learning to Play Strong Poker. In *Machines that Learn to Play Games*. Huntington,NY: Nova Science Publishers. pp.225-42.

Bodén, M., 2001. A guide to recurrent neural networks and back propagation. *The DALLAS project. Report from the NUTEK-supported project AIS-8, SICS. Holst: Application of data analysis with learning systems*, Citeseer, pp.1-10.

Brown, D.E. & Gordon, G., 2005. Terrain Based Prediction to Reduce the Search Area in Response to Insurgent Attacks. In *Attacks. The 10th International Command and Control Research and Technology Symposium*., 2005. pp.13-16.

Burns, K., 2006. Style in Poker. In *In IEEE Symposium on Computational Intelligence and Games (CIG 2006)*., 2006. pp.257-64.

Burns, M. & Pearl, J., 1981. Causal and Diagnostic Inferences: A Comparison of Validity. *Organizational Behaviour and Human Performance*, 28, pp.379-94.

Byrnes, J.P., 2001. *Cognitive Development and Learning in Instructional Contexts*. 2nd ed. Boston, MA: Alyn & Bacon.

Callan, R., 2003. *Artificial Intelligence*. Basingstoke, Hampshire, UK: Palgrave Macmillan.

Campbell, M., Haone, A.J. & Hsu, F.-h., 2002. Deep Blue. *Artificial intelligence*, 134, pp.57-83.

Carter, R.G. & Levine, J., 2007. An Investigation into Tournament Poker Strategy using Evolutionary Algorithms. In *IEEE Symposium on Computational Intelligence and Games, 2007 (CIG 2007).*, 2007. pp.117-24.

Chellapilla, K. & Fogel, D.B., 1999. Evolving Neural Networks to Play Checkers without Expert Knowledge. *IEEE Transactions on Neural Networks*, 10(6), pp.1382-91.

Clark, B., 2006. *The Dying Days of Las Vegas 1-5 Stud, Two Plus Two Internet Magazine.* [Online] Available at: http://www.twoplustwo.com/magazine/issue21/clark0906.html [Accessed July 2010].

Cowley, B., Charles, D., Black, M. & Hickey, R., 2009. Analyzing player behavior in pacman using feature-driven decision theoretic predictive modeling. In *IEEE Symposium on Computational Intelligence and Games, 2009 (CIG 2009).*, 2009. IEEE Press pp.170-77.

Cowling, P.I., 2006. Writing AI as Sport. In S. Rabin, ed. *AI Game Programming Wisdom 3*. Charles River Media. pp.89-96.

D'Silva, T. et al., 2005. Retaining Learned Behavior During Real-Time Neuroevolution. *Artificial Intelligence and Interactive Digital Entertainment*, pp.39-44.

Davidson, A., Billings, D., Schaeffer, J. & Szafron, D., 2000. Improved Opponent Modeling in Poker. In *Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI'2000)*., 2000. pp.1467-73.

De Jong, K.A., 1975. University Microfilms No. 76-09381 *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD Thesis. Ann Arbor, MI: The University of Michigan.

Deneve, S., 2008. Bayesian Spiking Neurons I: Inference. *Neural Computation*, 20, The MIT Press, pp.91-117.

Denzinger, J., Loose, K., Gates, D. & Buchanan, J., 2005. Dealing with parameterized actions in behavior testing of commercial computer games. In *IEEE Symposium on Computational Intelligence and Games (CIG '05)*. Colchester, UK, 2005. pp.51-58.

Egnor, D., 1999. *Iocaine Powder*. [Online] Available at: http://www.ofb.net/~egnor/iocaine.html [Accessed November 2008].

Elman, J.L., 1990. Finding Structure in Time. *Cognitive Science*, 14, Cognitive Science Society, pp.179-211.

Felix, D. & Reis, L., 2008. An Experimental Approach to Online Opponent Modeling in Texas Hold'em Poker. *Advances in Artificial Intelligence-SBIA 2008*, Springer, pp.83-92.

Findler, N., 1977. Studies in Machine Cognition Using the Game of Poker. *CACM*, 20(4), pp.230-45.

Finlay, J. & Dix, A., 1997. *An Introduction to Artificial Intelligence*. London: UCL Press.

Fraser, A.S., 1957. Simulation of genetic systems by automatic digital computers. I. Introduction. *Australian Journal of Biological Sciences*, 10, pp.484-91.

Funge, J., 2000. Cognitive modeling for games and animation. *Communications of the ACM*, 43, ACM, p.48.

Gal, Y. & Pfeffer, A., 2003. A language for modeling agents decision making processes in games. In *Autonomous Agents and Multi-Agents Systems Conference (AAMAS)*. Melbourne, Australia, 2003. ACM pp.265-72.

Gilpin, A. & Sandholm, T., 2006. Better automated abstraction techniques for imperfect information games, with application to Texas Hold em poker. In *Proceedings of the 6th National Conference on Artificial Intelligence (AAAI)*. Boston, MA, 2006. AAAI pp.1176--1183.

Gomez, F. & Miikkulainen, R., 1998. 2-D Pole Balancing with Recurrent Evolutionary Networks. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN-98)*. Skovde, Sweden, 1998. pp.425-30.

Greenwood, G., 2009. Deceptive strategies for the evolutionary minority game. In Lanzi, P.L., ed. *Proceedings of the 5th international conference on Computational Intelligence and Games (CIG '2009)*. Milan, Italy, 2009. IEEE Press pp.25-31.

Hamilton, S. & Garber, L., 1997. Deep Blue's Hardware-Software Synergy. *Computer*, 30(10), pp.29-35.

Hart, P.E., Nilsson, N.J. & Raphael, B., 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics SSC4*, pp.100-07.

Heijden, M.V., Bakkes, S. & Spronck, P., 2008. Dynamic Formations in Real-Time Strategy Games. In Hingston, P. & Barone, L., eds. *IEEE Symposium On Computational Intelligence and Games, 2008 (CIG '08).* Perth, WA, 2008. IEEE Press pp.47-54.

Hingston, P., 2009. A Turing Test for Computer Game Bots. *IEEE Transactions on Computational Intelligence and AI In Games*, 1(3), pp.169-86.

Hladky, S. & Bulitko, V., 2008. An Evaluation of Models for Predicting Opponent Positions in First-Person Shooter Video Games. *Computational Intelligence*, pp.39-46.

Holland, J.H., 1975. *Adaptation in Natural and Artificial Systems*. Ann Arbor, Michigan: University of Michigan Press.

Johanson, M. & Bowling, M., 2009. Data-biased robust counter strategies. In van Dyk, D. & Welling, M., eds. *Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS)*. Clearwater Beach, FL, 2009. Microtome Publishing pp.264-71.

Johnson-Laird, P.N., 1983. *Mental models: towards a cognitive science of language, inference and consciousness*. Cambridge, U.K.: Cambridge press.

Kaboli, A., Bowling, M. & Musilek, P., 2006. Bayesian calibration for Monte Carlo localization., 2006. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 p.964.

Kaminsky, R., Enev, M. & Andersen, E., 2008. *Identifying Game Players with Mouse Biometrics*. Technical Report. University of Washington.

Kendall, G. & Willdig, M., 2001. An investigation of an adaptive poker player. *AI 2001: Advances in Artificial Intelligence*, Springer, pp.217-29.

Kim, K.-j. & Cho, S.-b., 2008. Ensemble Approaches in Evolutionary Game Strategies: A Case Study in Othello. In Hingston, P. & Barone, L., eds. *IEEE Symposium On Computational Intelligence and Games, 2008 (CIG '08)*. Perth, WA, 2008. IEEE Press pp.212-19.

Kloetzer, J., Iida, H. & Bouzy, B., 2008. A Comparative Study of Solvers in Amazons Endgames. In Hingston, P. & Barone, P., eds. *IEEE Symposium On Computational Intelligence and Games, 2008 (CIG '08).* Perth, WA, 2008. IEEE Press pp.378-84.

Knuth, D.E. & Moore, R.W., 1973. An Analysis of Alpha-Beta Pruning. *Artificial Intelligence*, 6(4), pp.293-326.

Koller, D. & Pfeffer, A., 1995. Generating and solving imperfect information games. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*., 1995. pp.1185-92.

Koller, D. & Pfeffer, A., 1997. Representations and solutions for game-theoretic problems. *Artificial Intelligence*, 94(1), pp.167-215.

Krumm, J., 2006. Real Time Destination Prediction Based on Efficient Routes. In *Society of Automotive Engineers (SAE) 2006 World Congress*., 2006.

Kuhn, H.W., 1950. A simplified two-person poker. *Contributions to the Theory of Games*, pp.92-103.

Ku, K., Mak, M. & Siu, W., 2000. A study of the Lamarckian evolution of recurrent neural networks. *IEEE Transactions on Evolutionary Computation*, 4(1), IEEE Press, pp.31-42.

Laird, J.E., 2000. An Exploration into Computer Games and Computer Generated Forces. In *The Ninth Conference on Computer Generated Forces and Behavior Representation*. Orlando FL., 2000.

Laird, J.E., 2001. It Knows What You're Going to Do: Adding Anticipation to a Quakebot. In *Proceedings of the fifth international conference on Autonomous agents*. Montreal, Quebec, Canada, 2001. ACM Press pp.385-92.

Laird, J. & Arbor, A., 2000. *Design Goals for Autonomous Synthetic Characters*. Technical Report. University of Michigan.

Laird, J.E. & Duchi, J.C., 2000. Creating human-like synthetic characters with multiple skill levels: A case study using the soar quakebot. In *Simulating Human Agents, Papers from the 2000 AAAI Fall Symposium.* Menlo Park, CA, 2000. AAAI Press pp.75-79.

Laird, J. & Van Lent, M., 2001. Human-level AIs killer application: Interactive computer games. In *Proceedings of the 17th National Conference on Artificial Intelligence*. Menlo Park, CA, 2001. AAAI Press pp.1171-78.

Lanctot, M., Waugh, K., Zinkevich, M. & Bowling, M., 2009. Monte carlo sampling for regret minimisation in extensive games. *Advances in Neural Network Processing Systems 22 (NIPS)*, pp.1078-76.

Lawrence, S., Giles, C.L. & Tsoi, A.C., 1996. UMIACSTR 96-22 and CS-TR-3617 *What size neural network gives optimal generalization? Convergence properties of backpropagation*. Technical Report. University of Maryland.

Le Doux, J., 1997. *The Emotional Brain: The mysterious underpinnings of emotional life*. New york: Simon & Schuster.

Lee, K. & Mahajan, S., 1990. The development of a World-Class Othello program. *Artificial Intelligence*, (43), pp.21-36.

Lent, M.V. & Laird, J., 1998. Developing an Artificial Intelligence Engine. In *Proceedings of the 1999 Game Developers' Conference*.. San Jose, CA., 1998.

Leung, M.T., Engeler, W.E. & Frank, P., 1990. Fingerprint processing using backpropagation neural networks. In *Proceedings of the International Joint Conference on Neural Networks I*., 1990. pp.15-20.

Li, S., Chen, C. & Li, L., 2008. A new method for path prediction in network games. *Computers in Entertainment*, 5(4), pp.1-12.

Liu, Y., Comaniciu, C. & Man, H., 2006. A bayesian game approach for intrusion detection in wireless ad hoc networks. *Proceedings of the Workshop on Game Theory for Communications and Networks*, p.4.

Livingston, D., 2006. Turing's Test and Believable AI in Games. *ACM Computers in Entertainment (CIE)*, 4(1), pp.6-18.

Lockett, A., Chen, C. & Miikkulainen, R., 2007. Evolving Explicit Opponent Models in Game Playing. In *Opponent Models in Game Playing. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-07)*. San Francisco, 2007. Kaufmann pp.2106-13.

Lockett, A. & Miikkulainen, R., 2008. Evolving Opponent Models for Texas Hold'em. In *2008 IEEE Conference on Computational Intelligence in Games.* Perth, WA, 2008. pp.31--38.

Louis, S.J. & Miles, C., 2005. Combining Case-Based Memory with Genetic Algorithm Search for Competent Game AI. In *ICCBR Workshops*. Chicago, IL, USA, 2005. pp.193-205.

Lucas, S.M., 2008. Computational intelligence and games: Challenges and opportunities. *International Journal of Automation and Computing*, 5(1), pp.45-57.

Lucas, S.M. & Robles, D., 2009. A simple tree search method for playing Ms. Pac-Man. In Lanzi, P.L., ed. *IEEE Symposium on Computational Intelligence and Games 2009 (CIG 2009).* Milan, Italy, 2009. IEEE Press pp.249-55.

Magerko, B. et al., 2004. AI Characters and Directors for Interactive Computer Games. In *In Proceedings of the 2004 Innovative Applications of Artificial Intelligence Conference*. San Jose, CA, 2004. AAAI Press pp.877-84.

Marin, C., Castillo, L.P. & Garrido, L., 2005. Dynamic adaptive opponent modeling: Predicting opponent motion while playing soccer. In Alonso, E. & Guessoum, Z., eds. *Fifth European Workshop on Adaptive Agents and Multiagent Systems*. Paris, France, 2005.

Matthews, J., 2002. Basic A* Pathfinding Made Simple. In S. Rabin, ed. *AI Game Programming Wisdom*. Hingham MA: Charles River Media. pp.105-13.

McCulloch, W. & Pitts, W., 1943. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5, pp.115-33.

Mcpartland, M. & Gallagher, M., 2008. Creating a Multi-Purpose First Person Shooter Bot with Reinforcement Learning. In Hingston, P. & Barone, L., eds. *IEEE Symposium On Computational Intelligence and Games, 2008 (CIG '08).* Perth, WA, 2008. IEEE Press pp.143-50.

Mcquesten, P. & Miikkulainen, R., 1997. Culling and Teaching in Neuro-evolution. In *Proceedings of the Seventh International Conference on Genetic Algorithms*. East Lansing, MI, 1997. pp.760-67.

Mehta, M. & Corradini, A., 2009. Evaluation of a Domain Independent Approach to Natural Language Processing for Game-like User Interfaces. In Lanzi, P.L., ed. *Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG 2009)*. Milan, Italy, 2009. IEEE Press pp.225-32.

Meng, C. & Pakath, R., 2001. The Iterated Prisoner's Dilemma: Early Experiences with Learning Classifier System-based Simple Agents. *Decision Support Systems*, 31(4), pp.379-403.

Micheli, A., Sona, D. & Sperduti, A., 2004. Contextual Processing of Structured Data by Recursive Cascade Correlation. *IEEE Transactions on Neural Networks*, 15(6), IEEE Press, pp.1396-410.

Miikkulainen, R., Bryant, B.D., Cornelius, R. & Karpov, I.V., 2006. Computational Intelligence in Games. In G.Y. Yen & D.B. Fogel, eds. *Computational Intelligence: Principles and Practice*. Piscataway, NJ: IEEE Press. pp.155-91.

Miles, C. & Louis, S.J., 2005. Case-Injection Improves Response Time for a Real-Time Strategy Game. In Lucas, S. & Kendall, G., eds. *Proceedings of the 2005 IEEE Symposium on Computational Intelligence and Games (CIG05)*. Colchester, Essex, 2005. IEEE Press pp.149-56.

Millington, I., 2006. *Artificial Intelligence for Games*. San Francisco: Morgan Kaufmann.

Mitchell, M., 1996. *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press.

Mittal, S. & Deb, K., 2009. Optimal strategies of the iterated prisoner's dilemma problem for multiple conflicting objectives. *IEEE Transactions on Evolutionary Computation*, 13(3), IEEE Press, pp.554-65.

Mommersteeg, F., 2002. Pattern Recognition with Sequential Prediction. In S. Rabin, ed. *AI Game Programming Wisdom*. Clifton Park, NY: Charles River Media. pp.586-95.

Monroy, G.A. & Stanley, K.O., 2006. Coevolution of Neural Networks using a Layered Pareto Archive. In Keijzer, M., Cattolico, M., Arnold, D. & al., e., eds. *Proceedings of the 8th annual conference on Genetic and evolutionary computation (GECCO 06)*. Seattle, WA, 2006. ACM Press pp.329 - 336.

Moriarty, D.E. & Miikkulainen, R., 1994. Evolving Neural Networks To Focus Minimax Search. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94, Seattle, WA)*. Cambridge, MA, 1994. MIT Press pp.1371-77.

Moriarty, D.E. & Miikkulainen, R., 1995. Discovering Complex Othello Strategies Through Evolutionary Neural Networks. *Connection Science*, 7, pp.195-209.

Moriarty, D.E. & Miikkulainen, R., 1997. Forming Neural Networks Through Efficient And Adaptive Coevolution. *Evolutionary Computation*, 5(4), MIT Press, Cambridge, MA, pp.373-99.

Moriarty, D.E. & Miikkulainen, R., 1998. Hierarchical Evolution Of Neural Networks. In *In Proceedings of the 1998 IEEE Conference on Evolutionary Computation (ICEC98)*. Anchorage, AK, 1998. IEEE Press pp.428-33.

Moriarty, D., Miikkulainen, R. & Kaelbling, P., 1996. Efficient Reinforcement Learning through Symbiotic Evolution. *Machine Learning*, 22, pp.11-32.

Nakashima, T., Takatani, M., Ishibuchi, H. & Nii, M., 2006. The effect of using match history on the evolution of robocup soccer team strategies. In Louis, S.J., ed. *In proceedings of 2006 IEEE Symposium on Computational Intelligence and Games (CIG 2006)*. Reno/Lake Tahoe, 2006. IEEE Press pp.60-66.

Nash, J.F. & Shapley, L.S., 1950. A simple three-person poker game. *Contributions to the Theory of Games*, 1, pp.105-16.

Negnevitsky, M., 2005. *Artificial Intelligence (A guide to Intelligent Systems)*. 2nd ed. Harlow, Essex: Addison Wesley.

Nova Southeastern University, 2010. *Study Finds Poker Players Using Drugs to Enhance Performance*. [Online] Available at: http://www.sciencedaily.com/releases/2010/06/100601171840.htm [Accessed August 2010].

Papp, D.R., 1998. *Dealing with imperfect information in poker*. Master's Thesis, University of Alberta, Edmonton. University of Alberta Edmonton, Alberta, Canada.

Parker, M. & Bryant, B.D., 2009. Backpropagation without Human Supervision for Visual Control in Quake II. In Lanzi, P.L., ed. *Proceedings of the 5th international conference on Computational Intelligence and Games (CIG 2009)*. Milan, Italy, 2009. IEEE Press pp.287-93.

Picton, P., 2000. *Neural Networks*. 2nd ed. Basingstoke: Palgrave.

Ponsen, M., Lanctot, M. & de Jong, S., 2010. MCRNR: Fast Computing of Restricted Nash Responses by Means of Sampling. In *(To Appear) Interactive Decision Theory and Game Theory Workshop at the Twenty-Fourth Conference on Artificial Intelligence (AAAI-10)*., 2010. AAAI Press.

Ponsen, M.J., Muñoz-avila, H., Spronck, P. & Aha, D.W., 2005. Automatically Acquiring Domain Knowledge For Adaptive Game AI Using Evolutionary Learning. In *Seventeenth Conference on Innovative Applications of Artificial Intelligence*., 2005. AAAI Press pp.1535-40.

Ponsen, M., Munoz-avila, H., Spronck, P. & Aha, D., 2006. Automatically generating game tactics via evolutionary learning. *AI Magazine*, 27(3), AAAI Press, pp.75-84.

Pour, P.A. et al., 2008. Brain-Computer Interface : Next Generation Thought Controlled Distributed Video Game Development Platform. In Hingston, P. & Barone, L., eds. *IEEE Symposium on Computational Intelligence and Games (CIG '2008)*. Perth, WA, 2008. IEEE Press pp.251-57.

Randall, T.W.G., Cowling, P.I. & Baker, R.J.S., 2007. Learning Ship Combat Strategies in the Commercial Video Game DEFCON. In Rigas, D., ed. *8th Informatics Research Workshop for Research Students, University of Bradford, 2007.* Bradford, West Yorkshire, 2007. University of Bradford pp.182-83.

Randall, T.W.G., Cowling, P.I., Baker, R.J.S. & Jiang, P., 2009. Using Neural Networks for Strategy Selection in Real-Time Strategy Games. In *Proceedings of the AISB Symposium on AI & Games*. Edinburgh, UK, 2009.

Richards, N., Moriarty, D.E. & Miikkulainen, R., 1998. Evolving Neural Networks to Play Go. In Back, T., ed. *Proceedings of the Seventh International Conference on Genetic Algorithms*. East Lansing, MI, January-February 1998. Morgan Kaufmann pp.768-75.

Rosenblatt, F., 1959. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, 65, pp.386-408.

Rosenbloom, P.S., 1988. A World-Championship-Level Othello Program. *Computer Games II*, 2, Springer-VErlag, pp.365-408.

Rosin, C.D. & Belew, R.K., 1997. New methods for competitive coevolution. *Evolutionary Computation*, 5, MIT Press, pp.1--29.

Ross, S.M., 1971. Goofspiel - The Game of Pure Strategy. *Journal of Applied Probability*, 8(3), JSTOR, pp.621-25.

Russell, J. & Norvig, P., 1995. *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, NJ: Prentice Hall.

Sakaguchi, M. & Sakai, S., 1992. Solutions of Some Three-person Stud and Draw Poker. *Mathematics Japonica*, 6(37), pp.1147-60.

Saund, E., 2006. The Information Conveyed By Opponents' Betting Behavior in Poker. In Louis, S., ed. *In IEEE Symposium on Computational Intelligence and Games (CIG 2006)*. Reno/Lake Tahoe, 2006. IEEE Press pp.126-33.

Schaeffer, J. et al., 2007. Checkers is Solved. *Science Express*, 19 July. pp.1518-22.

Schaul, T. & Schmidhuber, J., 2008. A Scalable Neural Network Architecture for Board Games. In Hingston, P. & Barone, L., eds. *IEEE Symposium On Computational Intelligence and Games, 2008. CIG '08.* Perth, WA, 2008. IEEE Press pp.357-64.

Shea, P.M. & Liu, F., 1990. Operational experience with a neural network in the detection of explosives in checked airline baggage. In *Proceedings of the International Joint Conference on Neural Networks, Vol. II*., 1990. pp.175-78.

Simari, G., Sliva, A. & Nau, D., 2006. A stochastic language for modelling opponent agents. In *proceedings International Conference on Autonomous Agents and Multiagent Systems*. Hakodate, Japan, 2006. ACM Press pp.244-46.

Sklansky, D., 1992. *The Theory of Poker*. Two Plus Two Publishing.

Song, C., Qu, Z., Blumm, N. & Barabási, A.-L., 2010. Limits of Predictability in Human Mobility. *Science*, 327(5968), pp.1018-21.

Spronck, P., 2005. A model for reliable adaptive game intelligence. *in Proceedings of 2005 IJCAI Workshop on Reasoning, Representation, and Learning in Computer Games*, pp.95-100.

Spronck, P. & Herik, J.V., 2005. A tutoring system for commercial games. *Lecture Notes in Computer Science*, (3711), Springer-Verlag, pp.389-400.

Spronck, P., Ponsen, M. & Sprinkhuizen-kuyper, I., 2006. Adaptive Game AI with Dynamic Scripting. *Machine Learning*, 63(3), pp.217-48.

Spronck, P., Sprinkhuizen-Kuyper, I. & Postma, E., 2004. Difficulty Scaling of Game AI. In El Rhalibi, A. & Van Welden, D., eds. *GAME-ON 2004: 5th International Conference on Intelligent Games and Simulation*. Het Pand, Ghent, Belgium, 2004. pp.33-37.

Spronck, P., Sprinkhuizen-Kuyper, I. & Postma, E., 2004. Online Adaptation of Game Opponent AI with Dynamic Scripting. *International Journal of Intelligent Games and Simulation (eds. N.E. Gough and Q.H. Mehdi)*, 3(1), pp.45-53.

Spronck, P. et al., 2004. Enhancing the Performance of Dynamic Scripting in Computer Games. In *ICEC 2004*. Cambridge, UK, 2004. ACM Press pp.296-307.

Stanley, K.O., Bryant, B.D. & Miikkulainen, R., 2005. Evolving Neural Network Agents in the NERO Video Game. In Lucas, S. & Kendall, G., eds. *Proceedings of the IEEE 2005 Symposium on Computational Intelligence and Games (CIG '05)*. Colchester, Essex, 2005. IEEE Press pp.182-89.

Stanley, K. & Miikkulainen, R., 2002b. Continual Coevolution Through Complexification. In Langdon, W.B., Cant-Paz, E., Mathias, K.E. & al., e., eds. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002)*. San Francisco, 2002b. Kaufmann pp.113-20.

Stanley, K.O. & Miikkulainen, R., 2002. Efficient Evolution of Neural Network Topologies. In *Proceedings of the 2002 Congress on Evolutionary Computation*. Washington DC, 2002. IEEE Press pp.1757-62.

Stanley, K.O. & Miikkulainen, R., 2002. Efficient Reinforcement Learning through Evolving Neural Network Topologies. In Langdon, W.B., Cant-Paz, E., Mathias, K.E. & al., e., eds. *In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2002).* San Francisco, CA., 2002. Mogan Kaufmann pp.569-77.

Stanley, K.O. & Miikkulainen, R., 2002. Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation*, 10(2), pp.99-127.

Steffens, T., 2003. Feature-based declarative opponent modeling. In L. Iocchi, H. Matsubara, A. Weitzenfeld & C. Zhou, eds. *RoboCup 2003: Robot Soccer World Cup VII*. Springer. pp.125-36.

Stensrud, B.S. & Gonzalez, A.J., 2008. Discovery of High-Level Behavior From Observation of Human Performance in a Strategic Game. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 38(3), IEEE Press, pp.855-74.

Suffecool, K., 2006. *Cactus Kev's Poker Hand Evaluator*. [Online] Available at: http://www.suffecool.net/poker/evaluator.html [Accessed August 2009].

Terry, M. & Mihok, B., n.d. A Bayesian Net Inference Tool for Hidden State in Texas Hold em Poker. *Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology*.

Tesauro, G., 1990. Neurogammon: a neural-network backgammon program. In *1990 IJCNN International Joint Conference on Neural Networks*. San Diego, CA, USA., 1990. IEEE Press pp.33-39.

Tesauro, G., 1995. Temporal Difference Learning and TD-Gammon. *Communications of the ACM*, 38(3), ACM Press, pp.58-68.

The Guardian, 2007. *Computer program takes draughts crown*. [Online] Available at: http://www.guardian.co.uk/technology/2007/jul/20/news.uknews [Accessed October 2008].

Thompson, T., Levine, J. & Wotherspoon, R., 2008. Evolution of Counter-Strategies: Application of Co-evolution to Texas Hold 'em Poker. In Hingston, P. & Barone, L., eds. *IEEE Symposium On Computational Intelligence and Games, 2008 (CIG '08).* Perth, WA, 2008. IEEE Press pp.16-22.

Thompson, T., Mcmillan, L., Levine, J. & Andrew, A., 2008. An Evaluation of the Benefits of Look-Ahead in Pac-Man. In Hingston, P. & Barone, L., eds. *IEEE Symposium On Computational Intelligence and Games (CIG '2008)*. Perth, WA, 2008. IEEE Press pp.310 - 315.

Towell, G. & Shavlik, J., 1994. Knowledge-based artificial neural networks. *Artificial Intelligence*, 70(1-2), pp.119-65.

Tu, X., Funge, J. & Terzopoulos, D., 1999. Cognitive modeling: Knowledge, reasoning and planning for intelligent characters. In ACM, ed. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques. 8-13 August 1999*. Los Angeles, CA, 1999. ACM pp.29-38.

Turing, A., 1950. Computing Machinery and Intelligence. *Mind*, 236, pp.433-60.

Tversky, A. & Kahneman, D., 1982. Causal Schemes in Judgements Under Uncertainty. In P. Slovic & A. Tversky, eds. *Judgements Under Uncertainty: Heuristics and Biases*. New York: Cambridge University Press. pp.117-28.

van den Herik, H.J. & Donkers, H.S.P., 2005. Opponent Modelling and Commercial Games. In G. Kendall and S. Lucas, ed. *Proceedings of IEEE 2005 Symposium on Computational Intelligence and Games CIG'05*. Colchester, UK, 2005. IEEE Press pp.15-25.

Varoufakis, Y., 2001. General introduction: Game theory's quest for a single, unifying framework for the social sciences. In Y. Varoufakis, ed. *Game Theory: Critical Concepts in the Social Sciences*. London: Routledge.

von Neumann, J. & Morgenstern, O., 1944. *Theory of Games and Economic Behavior*. Princeton Univ. Press.

Waugh, K. et al., 2009. A practical use for imperfect recall. In Bulitko, V. & Beck, J.C., eds. *Proceedings of the Eighth Symposium on Abstraction, Reformulation and Approximation (SARA 2009)*. Menlo Park, CA, 2009. AAAI Press p.175–182.

Weber, B.G. & Mateas, M., 2009. A Data Mining Approach to Strategy Prediction. In Lanzi, P.L., ed. *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on 7-10 Sept*. Milan, 2009. IEEE Press pp.140-47.

Whiteson, S. et al., 2005. Automatic feature selection in neuroevolution. In Beyer, H.-G. & O'Reilly, U.-M., eds. *GECCO 2005: Proceedings of the Genetic and Evolutionary Computation Comference*. Washington DC, 2005. IEEE Press pp.1225-32.

Yao, X., 1999. Evolving Artificial Neural Networks. *Proceedings of the IEEE*, 87(9), pp.1423-47.

Yao, X. & Islam, M., 2008. Edward P.K. Tsang and Serafin Martinez-Jaramillo Centre for Computational Finance and Economic Agents (CCFEA). *IEEE Computational Intelligence Magazine*, IEEE Press, pp.31-42.

Yao, X. & Liu, Y., 1998. Making Use of Population Information in Evolutionary Artificial Neural Networks. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 28(3), pp.417-25.

Yong, C.H. & Miikkulainen, R., 2001. AI07-338 *Cooperative Coevolution Of Multi-Agent Systems*. Technical Report. Austin, Texas: The University of Texas at Austin.

Zhao, X., Xu, R. & Kwan, C., 2004. Ship-motion prediction: algorithms and simulation results. In *IEEE International Conference on Acoustics, Speech, and Signal Processing. (ICASSP '04) 17-21 May 2004*. Montreal, Quebec, Canada, 2004. IEEE Press pp.125-28.

Zinkevich, M., Johanson, M., Bowling, M. & Piccone, C., 2008. Regret Minimization in Games with Incomplete Information. *Advances in Neural Information Processing 21 (NIPS)*.