Online Agent Modelling in Human-Scale Problems

by

Nolan Deans Carlisle Bard

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

University of Alberta

Abstract

Ideal agent behaviour in multiagent environments depends on the behaviour of other agents. Consequently, acting to maximize utility is challenging since an agent must gather and exploit knowledge about how the other (potentially adaptive) agents behave. In this thesis, we investigate how an agent can efficiently tailor its behaviour to other agents during interaction in order to maximize its performance. This thesis presents three main contributions.

First and foremost, the thesis characterizes and contrasts the traditional agent modelling approach — where practitioners explicitly estimate and subsequently respond to a generative model of an agent's behaviour — with an alternative approach called *implicit modelling*. Using traditional explicit modelling in complex human-scale domains is difficult since an agent must efficiently estimate sophisticated behaviours from observations that may be stochastic and partially observable. Even after estimating a generative model, it may be impractical to compute a response that is robust to modelling error during interaction. The implicit modelling framework avoids many of these challenges by estimating the utilities of a *portfolio of strategies*. Furthermore, implicit modelling naturally affords the opportunity to generate the portfolio offline, which provides practitioners with the time necessary for computationally expensive robust response techniques. We introduce an end-to-end approach for building an implicit modelling agent and empirically validate it in several poker domains.

Second, the thesis contributes the first empirical analysis of how the granularity of an agent's representation of a multiagent environment — including its beliefs about the other agents — impacts two common objectives: performance against suboptimal agents and robustness against worst-case agents. We show that using *asymmetric representations* allows for practitioners to trade off these objectives whereas commonplace symmetric representations optimize neither.

Third, we contribute a novel *decision-theoretic clustering* algorithm. While many existing clustering techniques optimize for spatial similarity between objects, we demonstrate that such spatial clustering can fail to capture similarity in how an agent should respond to the clusters to maximize utility. Our algorithm exploits structure in the utility function to allow for an efficient greedy approximation to this computationally hard optimization. We prove worst-case approximation bounds for our algorithm and empirically validate the approach by clustering agent behaviours in extensive-form games.

These three contributions provide practitioners with a foundation of practical techniques for constructing an effective portfolio of strategies and using the portfolio to adapt an agent's behaviour. Our empirical evaluation of implicit modelling agents in a variety of poker games demonstrates that implicit modelling is an effective agent modelling approach for online real-time adaptation in complex human-scale domains.

To ask may be but a moment's shame,
not to ask and remain ignorant is a lifelong shame.
Kanō Jigorō

It is not important to be better than someone else,
but to be better than yesterday.
Kanō Jigorō

Acknowledgements

As I reach the "destination" of my graduate studies, I am afforded a chance to reflect on and thank the vast community that has been part of my long journey. I have been incredibly fortunate to share this trip with so many amazing people, each of whom has contributed to supporting, inspiring, and guiding me along the way.

First, I could not have asked for a better mentor than my supervisor, Michael Bowling. He supported me throughout my studies – both as a patient adviser and jovial friend – and continues to inspire me with his brilliance. I hope that, one day, I will pass down the same quality of mentorship that he gave to me.

To the other members of my committee, Robert Holte, Dale Schuurmans, Martin Müller, and Peter Stone, I thank you for your valuable guidance and insightful feedback.

The members of the Computer Poker Research Group at the University of Alberta have been instrumental to this research, both as friends and colleagues. Their perspicacity, enthusiasm, teamwork, and camaraderie helped drive me and this research. It has been my privilege to be part of this exciting team, the research and competitions it fosters, and its achievements. Thank you Duane Szafron, Robert Holte, Jonathan Schaeffer, Neil Burch, Martin Zinkevich, Chris Archibald, Viliam Lisý, Kevin Waugh, Morgan Kan, Richard Gibson, Nick Abou Risk, Josh Davidson, John Hawkin, Dave Schnizlein, Darse Billings, Trevor Davis, Dustin Morrill, Parisa Mazrooei, and Bryce Paradis. I also want to particularly thank Michael Johanson, who played a significant role throughout my studies; whether it was collaborating on a paper, research discussions over beer, partaking in (or organizing) social events, or keeping me sane during writing, he was there.

To the many faculty, staff, and students in the Department of Computing Science, thank you for building an inclusive community where everyone is encouraged to collaborate and socialize. My time in the department afforded numerous CSGSA and UACS events, games parties, defence beers, ski trips, and Turkey Trots. This cordial environment gave me the opportunity to build friendships with some amazing people: Andrew Butcher, Jag Grewal, Vanessa Burke, William Thorne, Jeff Ryan, Curtis Onuczko, Alona Fyshe, Dan Lizotte, Brad Joyce, Jess Enright, Richard Valenzano, Jacqueline Smith, Paul Berube, Marc Bellemare, Jeff Siegel, Anna Koop, Leah Hackman, Shayna Bowling, Kit Chen, Barry Gergel, Sheehan Khan, Marc Lanctot, John Arnold, Brian Tanner, David Thue, Adam White, Martha White, Joel Veness, Dave Churchill, Levi Lelis, and Marlos Machado. Furthermore, I would like to thank my longtime friends Mark Lee and the late Dirk Henkemans, whose infectious enthusiasm about computing science helped lead me into the discipline so many years ago.

Many of the friends that I met through the department also played pivotal roles in drawing me into physical activities that have become central to my life. In particular, Ryan Warden and Steven Roehr graciously introduced me to weight training when I was an undergrad, and Chris Rayner encouraged me to start judo near the start of my Ph.D. program. Training has enhanced my time as a graduate student, helped keep me sane, and led me to meet other remarkable friends and mentors. The sensei at the University of Alberta's judo club – Gord Okamura, Kelly Palmer, and formerly Ron Senda – have been inspiring and encouraging, despite my overreliance on sumi gaeshi. It has also been my pleasure to throw and be thrown by fellow judoka Joe Parrell, Jason Woo, Amin Moazeni, Behzad Vafaeian, Nathan Deisman, David Moulds, and Tobias Weingartner, amongst others. My time at the gym also introduced me to friends and fellow aficionados of lifting heavy things up and putting them back down: Brendan Woychuk, Matt Radkie, Adam Butson, Emily Butson, Dion Pasichnyk, and Minkyu Kang.

Most of all, I want to thank my parents, Jeanne Byron and Forrest Bard, my brother, Ryan Bard, and all of my extended family for their love and support. All of you have been part of my journey from the beginning and have supported me, regardless of my quirks. I could not have chosen a better family.

Contents

1 Introduction 1

1.1 Objective 1

1.2 Agent Modelling 3

1.3 Contributions 3

1.4 Thesis Outline 4

2 Background 5

2.1 Extensive-form games 5

2.2 Poker 7

2.2.1 Kuhn Poker 8

2.2.2 Leduc Hold'em 9

2.2.3 Texas Hold'em 9

2.3 Nash Equilibrium Approximation 10

2.3.1 Sequence Form and Linear Programming 10

2.3.2 Counterfactual Regret Minimization 11

2.4 Robust Responses 13

2.4.1 ε-safe Responses 14

2.4.2 Restricted Nash Responses 14

2.4.3 Data Biased Responses 15

2.5 Abstraction 16

2.5.1 Card Abstraction 17

2.5.2 Betting Abstraction 20

2.6 Multi-armed Bandits 21

2.6.1 Stochastic Bandits 21

2.6.2 Adversarial (Non-Stochastic) Bandits 22

2.7 Variance Reduction for Strategy Evaluation 23

2.7.1 Duplicate 24

2.7.2 Importance Sampling 24

2.7.3 Imaginary Observations 25

2.8 Submodular Optimization 26

2.8.1 Maximum Coverage Problems 26

3 Agent Modelling 28

3.1 Explicit Modelling 28

3.1.1 Learning an Explicit Model 29

3.1.2 Using an Explicit Model 31

3.2 Implicit Modelling 35

3.2.1 Using a Portfolio 36

3.2.2 Building a Portfolio 38

4 Asymmetric Abstractions 41
4.1 Nash Equilibrium Approximation 43
4.1.1 Experimental Design 43
4.1.2 Empirical Results 44
4.2 Robust Counter-Strategies 46
4.2.1 Experimental Design 47
4.2.2 Empirical Results 48
4.3 Summary 52

5 Decision-Theoretic Clustering 53
5.1 Segmentation Problems 54
5.2 Exploiting Structured Utility 57
5.2.1 Response Oracles 57
5.2.2 A Greedy Heuristic 58
5.3 Theoretical Results 59
5.4 Strategy Clustering 61
5.4.1 Experimental Design 61
5.4.2 Empirical Results 61
5.5 Summary 65

6 Human-scale Implicit Modelling 66
6.1 Case Study: Heads-up Limit Texas Hold'em 67
6.1.1 Agent Design 67
6.1.2 Empirical Results 69
6.2 Annual Computer Poker Competition Results 73
6.2.1 Heads-up Limit Texas Hold'em 74
6.2.2 3-player Limit Texas Hold'em 79
6.2.3 Heads-up No-limit Texas Hold'em 82
6.3 Summary 90

7 Conclusions and Future Work 91
7.1 Contributions 91
7.2 Directions for Future Work 93
7.2.1 Moving Beyond Poker 93
7.2.2 Efficient Strategy Evaluation 93
7.2.3 Online Adaptation 94
7.2.4 Robust Responses 94
7.2.5 Evaluating Adaptive Agents 96
7.3 Summary 96

Bibliography 97

A Annual Computer Poker Competition Results 104

B Annual Computer Poker Competition Agents 126
B.1 Heads-up Limit Texas Hold'em 128
B.1.1 2012 128
B.1.2 2013 129
B.1.3 2014 130
B.2 3-player Limit Texas Hold'em 131
B.2.1 2013 & 2014 131
B.3 Heads-up No-limit Texas Hold'em 132
B.3.1 2013 132
B.3.2 2014 134

List of Tables

4.1 Sizes of percentile abstractions 44
4.2 Comparison of Nash equilibria strategies using asymmetric abstractions . . . 45

6.1 Case study results versus 2011 ACPC heads-up limit hold'em TBR agents . . 72
6.2 2012 ACPC heads-up limit hold'em TBR: average winnings 75
6.3 2013 ACPC heads-up limit hold'em TBR: average winnings 76
6.4 2013 ACPC heads-up limit hold'em: Hyperborean agents 77
6.5 2014 ACPC heads-up limit hold'em TBR: average winnings 78
6.6 2013 ACPC 3-player limit hold'em TBR: average winnings 80
6.7 2013 ACPC 3-player limit hold'em: Hyperborean agents 80
6.8 2014 ACPC 3-player limit hold'em TBR: average winnings 81
6.9 2014 ACPC 3-player limit hold'em: Hyperborean agents 81
6.10 2013 ACPC heads-up no-limit hold'em TBR: average winnings 84
6.11 2013 ACPC heads-up no-limit hold'em: Hyperborean agents 86
6.12 2014 ACPC heads-up no-limit hold'em TBR: average winnings 88
6.13 2014 ACPC heads-up no-limit hold'em: Hyperborean agents 89

A.1 2012 ACPC heads-up limit hold'em TBR: uncapped winnings 105
A.2 2013 ACPC heads-up limit hold'em TBR: capped winnings 106
A.3 2013 ACPC heads-up limit hold'em TBR: statistical significance 107
A.4 2013 ACPC heads-up limit hold'em TBR: uncapped winnings 108
A.5 2013 ACPC heads-up limit hold'em TBR: uncapped statistical significance . . 109
A.6 2014 ACPC heads-up limit hold'em TBR: capped winnings 110
A.7 2014 ACPC heads-up limit hold'em TBR: statistical significance 111
A.8 2014 ACPC heads-up limit hold'em TBR: uncapped winnings 112
A.9 2014 ACPC heads-up limit hold'em TBR: uncapped statistical significance . . 113
A.10 2013 ACPC 3-player limit hold'em TBR: capped winnings 114
A.11 2013 ACPC 3-player limit hold'em TBR: statistical significance 115
A.12 2014 ACPC 3-player limit hold'em TBR: capped winnings 116
A.13 2014 ACPC 3-player limit hold'em TBR: statistical significance 117
A.14 2013 ACPC heads-up no-limit hold'em TBR: capped winnings 118
A.15 2013 ACPC heads-up no-limit hold'em TBR: statistical significance 119
A.16 2013 ACPC heads-up no-limit hold'em TBR: uncapped winnings 120
A.17 2013 ACPC heads-up no-limit hold'em TBR: uncapped statistical significance 121
A.18 2014 ACPC heads-up no-limit hold'em TBR: capped winnings 122
A.19 2014 ACPC heads-up no-limit hold'em TBR: statistical significance 123
A.20 2014 ACPC heads-up no-limit hold'em TBR: uncapped winnings 124
A.21 2014 ACPC heads-up no-limit hold'em TBR: uncapped statistical significance 125

List of Figures

2.1 Extensive-form game example: the “Mystery Box” game 6
2.2 Counterfactual value terms 12
2.3 Solving games via abstraction 17

3.1 Explicit modelling workflow 28
3.2 Strategies with Pareto optimal trade-off between model exploitation and worst-case performance 33
3.3 Implicit modelling workflow 36

4.1 Abstraction of the first round of Texas hold’em poker, dividing 1326 hands into 5 percentile $E[HS^2]$ buckets. 43
4.2 Impact of quantity of observations and model abstraction on exploitation of tilted 8-bucket equilibria 48
4.3 Impact of counter-strategy and opponent response abstraction size on RNR one-on-one and worst-case performance in the unabstracted game 50
4.4 Impact of opponent response abstraction size and quantity of observations on DBR one-on-one and worst-case performance in the unabstracted game . . . 51
4.5 Impact of opponent model abstraction size on one-on-one and worst-case performance in the unabstracted game with 100,000 observations 51

5.1 Rock-paper-scissors strategy simplex partitioned into best response regions . 54
5.2 Matrix view of a segmentation problem 56
5.3 Clusterings of player two Kuhn poker agents 63
5.4 Performance of different clustering techniques in toy poker domains 64

6.1 Illustration of the implicit modelling process 70
6.2 Case study results versus mimics with responses in the small portfolio 71
6.3 Case study results versus mimics with responses in the large portfolio 72
6.4 Impact of the expected winnings cap on the average bankroll of top agents from the 2013 heads-up no-limit ACPC 85
6.5 Impact of the expected winnings cap on the average bankroll of top agents from the 2014 heads-up no-limit ACPC 89

Chapter 1

Introduction

The world is replete with problems involving interactions between multiple agents. While humans have historically been the primary actors in these multiagent domains, the rise of artificial intelligence has been driving computer agents to the forefront of many multiagent problems. Even now, computer agents have been deployed for everything from relatively innocuous tasks, like automated phone systems and elevator control, to potentially life altering roles, such as autonomous vehicles and stock trading agents. This ubiquity makes computer agents central actors with a very real impact on the day-to-day lives of people around the world. Moreover, the growing reach of autonomous computer agents has made coping with the presence of other decision makers a key challenge for artificial intelligence.

An **agent**[1] is any autonomous decision maker — human or computer. In multiagent environments, agents must interact with other agents that may have unknown goals and capabilities. As such, an agent's ideal behaviour depends on how the other agents act. One common approach to designing agents in a multiagent system is to have an agent behave in a way that maximizes their performance assuming worst-case behaviour by the other agents. Though this approach is useful in many settings, the resulting agent behaviour is not ideal when the other agents do not conform to such pessimistic assumptions. An **ideal agent** must behave to maximize its performance with respect to the other agents at hand.

Learning about other agents and adapting to their behaviour is essential for such an ideal agent and is the central task of *agent modelling*[2]. This dissertation focuses on the problem of modelling agents during interaction in complex human-scale domains. Although agent modelling has been studied previously, practitioners face numerous practical challenges in using traditional agent modelling approaches in this context.

1.1 Objective

In this dissertation, we address a fundamental question in the field of agent modelling.

> In complex multiagent domains, how can an agent efficiently tailor its behaviour to other agents during interaction in order to maximize its performance?

An ideal agent must learn and exploit knowledge of how **other agents** in the environment act. We are interested in multiagent domains with other agents that are *external* (*i.e.*, agents that are outside our control). As such, an agent designer cannot (safely) make any assumptions about the behaviour of these agents.

[1] Since this work focusses on multiagent domains in the context of extensive-form games, we will also refer to agents as *players* and *opponents*.

[2] Substantial agent modelling research has also fallen under the banner of *opponent modelling*. Although this thesis investigates agent modelling in an adversarial setting, the contributions of this work are general enough that they could be employed in the broader context of arbitrary agents.

We seek to adapt to the other agents **during interaction** (*i.e.*, **online**). In this online learning setting, agents must learn and adapt while they accumulate reward throughout the interaction. Consequently, agents face a familiar *exploration-exploitation trade-off* where they must balance learning about the other agents with acting effectively themselves.

We examine this online learning problem in **complex domains**. These domains have an enormous number of distinct states. As a result, it is typically impractical for an agent to distinguish between every possible state and they must employ some form of approximation to act effectively. We focus on **human-scale** domains that offer sufficient complexity for humans to actively engage and compete in them.

In addition to a domain's size, other properties present additional challenges for agent modelling practitioners. We introduce five characteristics of our domains and the challenges they present.

- **Limited observations**

  Typically we need to model agents quickly: learning and adapting within a small number of observations. This is especially true when interacting with human agents as it is often impractical to observe an individual for a prolonged time.

- **Stochastic observations**

  Stochastic observations occur due to an agent or the environment acting according to some probability distribution over the possible actions. This tends to make it more difficult to discover the signal (*e.g.*, agent behaviour) within the noise of stochastic sampling.

- **Imperfect information**

  With imperfect information, the environment is only *partially observable.* That is, part of the state of the environment is hidden to our agent. If other agents condition their behaviour on information that our agent never gets to observe, then our agent may conflate otherwise distinct behaviours.

- **Maximizing versus satisficing**

  An agent's objective may only require them to sufficiently satisfy a given criterion in order to be "successful". For example, in the context of stock trading, an agent could be considered successful if they secure a desired number of shares for no more than some specified cost. However, agents face a much more nuanced decision making problem when outcomes have a range of quality, such at the cost of purchasing the shares.

- **Dynamic behaviour**

  Other agents may change their behaviour over time, potentially diminishing the value of prior observations. Examples of this include customers changing their purchasing habits over time or a game player changing their behaviour over the course of one or more matches.

These characteristics pose challenges for practitioners not only individually, but also through their combined effect. For example, a perceived change in an agent's behaviour could be attributed to any of stochastic observations, imperfect information, or dynamic behaviour. Modelling agents in domains with combinations of these features is a difficult task: a challenge only aggravated further when learning and adaptation must occur within a limited number of observations.

1.2 Agent Modelling

The traditional approach to agent modelling is to learn a generative model of the agent's action selection policy though observing its behaviour. This could involve directly estimating the agent's probability of selecting an action at every decision point, or estimating some parameters of a model that describes how the agent generates its behaviour. We call such a model an **explicit agent model**.

Explicit models provide a straightforward way to represent a rich space of agent behaviour. Although explicit models may be effective for small domains or when the space of reasonable agent behaviour is small and easily measurable, in complex systems they can present several practical challenges. First, rich explicit models for complex domains have a high-dimensional parameterization. Without substantial prior knowledge, building an accurate model can require a prohibitively large number of observations. Furthermore, in domains with imperfect information, inferring the parameters may be intractable. Finally, even given an accurate model, it is not obvious how to use this knowledge when responding. One possible approach would be to compute a response that maximizes the agent's utility given the model parameters. Johanson and colleagues (Johanson, Zinkevich, and Bowling 2008) showed that these best responses can be brittle and vulnerable to model error. Instead, they proposed a technique to generate robust responses that maximize utility given the model parameters, but subject to a measure of resilience against errors in the model. Unfortunately, robust responses are currently impractical to compute quickly online for complex systems.

Through learning and exploiting a generative model, explicit models provide one route towards achieving ideal agent behaviour. When agent performance is truly the goal — as opposed to producing the generative model itself — learning such an explicit model online is unnecessary.

1.3 Contributions

The main contributions of this thesis are divided into the following three parts.

Implicit agent modelling. We investigate a largely overlooked paradigm for agent modelling called **implicit modelling**. Instead of using online observations to estimate a generative model of the other agents' behaviour, implicit modelling summarizes their behaviour through the *utilities* of a **portfolio** of responses. Implicit models confer two key benefits: they remove the need to estimate a generative model online in order to produce a response, and they provide a natural representation for using responses that have been generated offline. These benefits are vital in complex domains where efficiently estimating a generative model online is challenging and the cost of computing responses, especially robust responses, can vastly exceed the time available to act. We present a novel characterization of agent modelling techniques that contrasts the explicit and implicit modelling approaches, highlighting the benefits and challenges of each. Furthermore, we address several practical challenges in using implicit models and empirically validate our approach relative to baseline approaches through a detailed case study. Finally, we empirically demonstrate the efficacy of this paradigm in human-scale domains by evaluating implicit modelling agents against competitors from several prior Annual Computer Poker Competitions — the premier venue for computer poker research. Part of this work was published as "Online Implicit Agent Modelling" (Bard et al. 2013).

The main question raised by using an implicit model is how one should generate a portfolio. Our remaining two contributions introduce general techniques for constructing responses that improve upon existing techniques and, in turn, result in more effective portfolios.

Asymmetric abstractions. Agents typically require some form of approximation or abstraction to act effectively in complex domains. In multiagent domains, these abstraction choices affect an agent's beliefs about the capabilities and behaviours of other agents. The standard approach when abstracting is to use symmetric abstraction: where all agents are assumed to distinguish states in the same way. We empirically evaluate the impact that abstraction choices have on agent performance, demonstrating both the benefits and potential pitfalls of using *asymmetric* abstractions instead of standard symmetric abstractions. Furthermore, we show that combining asymmetric abstractions with robust response techniques can produce responses which dominate their symmetric abstraction counterparts in terms of both exploitative power and worst-case utility. This work was published as "Asymmetric Abstractions for Adversarial Settings" (Bard, Johanson, and Bowling 2014).

Decision-theoretic clustering. Given prior knowledge of some agents (*e.g.*, observations from past interactions), one could construct a portfolio by generating a response to each agent. However, when the number of agents is large, such individual personalization becomes problematic. A large portfolio not only makes the online utility estimation of implicit modelling more difficult, but many of the responses could be very similar. To mitigate these problems while preserving a useful diversity of responses, we turn to clustering to identify groups of similar agents that we can focus on for response construction. Traditional clustering techniques would typically aim to group the agents together based on a distance metric, where a desirable clustering is one where the agents in a cluster are spatially close together. Instead, we desire to cluster based on *actionability*: the capacity for the clusters to suggest how we should construct responses that maximize utility with respect to the agents. Segmentation problems examine this decision-theoretic clustering task. Although finding optimal solutions to these problems is computationally hard, greedy-based approximation algorithms exist. However, in settings like poker where the agent has a combinatorially large number of candidate responses whose utilities must be considered, these algorithms are often intractable. We show that in many cases the utility function can be factored to allow for an efficient greedy algorithm even when there are exponentially large response spaces. This work was published as "Decision-theoretic Clustering of Strategies" (Bard et al. 2015).

1.4 Thesis Outline

We present the contributions of this thesis as follows. Chapter 2 introduces background material including extensive-form games, which provide a general model for sequential decision making in multiagent interactions; the specific poker domains used for our experimentation, including Texas hold'em poker; and other prior work essential to the contributions of the thesis. Chapter 3 compares the explicit and implicit modelling approaches, including prior explicit modelling efforts in poker and challenges facing the implicit modelling approach. Chapter 4 presents our analysis of asymmetric abstractions and how they can be used to improve the quality of responses. Chapter 5 introduces the decision-theoretic clustering problem, our greedy algorithm, and both theoretical and empirical results validating our approach. Chapter 6 provides an end-to-end description of how we apply implicit modelling to construct an adaptive poker agent: from building robust responses, selecting responses for the portfolio, and using online learning algorithms to identify the portfolio's ideal response during online interaction. Furthermore, it provides empirical evaluation validating the approach using agents from several prior years of the Annual Computer Poker Competition's total bankroll events. Finally, Chapter 7 concludes the thesis with several open challenges and possible avenues for future work on the implicit modelling framework, followed by some closing remarks.

Chapter 2

Background

In this chapter, we introduce central concepts and prior work used throughout this thesis. While the principles underpinning our agent modelling contributions can be applied quite broadly, we focus our investigation on extensive-form games and poker games in particular. Section 2.1 introduces the extensive-form game model and Section 2.2 describes the family of poker games used for our experimental domains. Sections 2.3 and 2.4 present prior work on techniques for creating strategies in extensive-form games, some of which we will build on. Human-scale games are often so large that they require some form of abstraction for these techniques to be feasible. In Section 2.5, we present some common poker abstraction techniques. Finally, the remainder of the chapter discusses background work that we build upon in our implicit modelling contributions: multi-armed bandit algorithms (Section 2.6), variance reduction for strategy evaluation (Section 2.7), and submodular optimization (Section 2.8).

2.1 Extensive-form games

Extensive-form games provide a general model for interactions between multiple agents making a sequence of decisions. An intuitive view of an extensive-form game is that of a rooted tree. To help illustrate this model, we introduce and describe a simple toy game in Figure 2.1 that we dub the "Mystery Box". The tree's nodes represent **histories** (*i.e.*, sequences) of actions, $h \in H$. Leaves of the tree are **terminal histories** $Z \subseteq H$ that represent the end of the game. Each terminal history $z \in Z$ has an associated **utility** (*i.e.*, reward or payoff) $u_i(z)$, for each **player** $i \in N = \{1, \ldots, n\}$. At each non-terminal history, h, an edge represents an **action** $a \in A(h)$ available to the **acting player** $P(h) \in N \cup \{c\}$, where c denotes **chance**. When $P(h) = c$ it is chance's turn to act, and chance selects action a according to a known fixed probability $\sigma_c(a|h)$. If history h begins with h', we call h' a **prefix** of h and denote it with $h' \sqsubseteq h$.

The histories where player i is acting (*i.e.*, when $P(h) = i$) are partitioned into **information sets** $I \in \mathcal{I}_i$ representing sets of histories that player i cannot differentiate. In **perfect information** games, players can observe past actions taken by all other players (including chance) and the information set partition consists of singletons. However when this is not the case, the game has **imperfect information** and at least some information sets will contain multiple histories. For example, in the "Mystery Box" game, players are unable to observe chance's action, which decides the contents of the box; Figure 2.1 depicts the resulting information sets as a dashed line around the blocks of the partition. For any two histories $h, h' \in I$, $A(h)$ must equal $A(h')$ and therefore we simply denote the actions available at an information set as $A(I)$. For a given history h, we denote the information set containing h as I_h. If for each player i, and each information set $I \in \mathcal{I}_i$, all histories

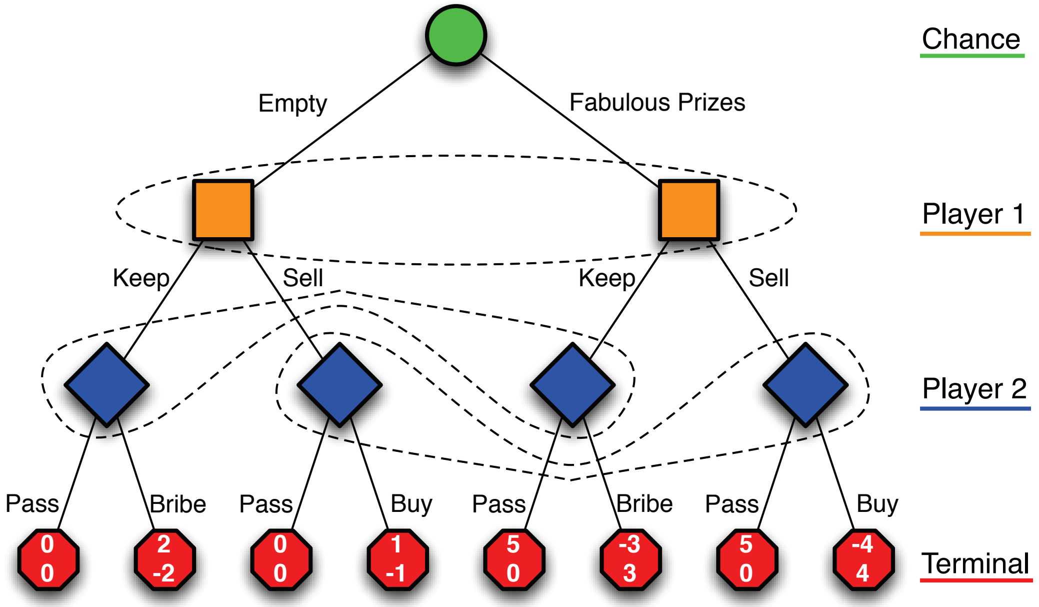


Figure 2.1: The extensive-form game tree for the "Mystery Box" game. The game begins with chance creating a "mystery box" that is either empty (worth \$0) or filled with prizes worth \$5. Player one then chooses either to keep the box or offer it for sale. After player one acts, player two can either pass, or pay player one to acquire the box. If player one opted to sell the box, player two can buy it for \$1, otherwise they must pay player one \$2 to"bribe" them for the box. Tiers of the game are labelled with the acting player.

$h \in I$ share a unique sequence of past player i information set and action pairs (*i.e.*, players never forget anything), the game is said to have **perfect recall**.

During the course of a game, each player selects actions according to their strategy (*i.e.*, policy). Formally, a **behavioural strategy** (henceforth a **strategy**) for player i, σ_i, is a function that maps each information set $I \in \mathcal{I}_i$ to a probability distribution over the legal actions $A(I)$. Note that this means players must act based solely on their current information set, thereby acting the same at all the indistinguishable game states within the information set. The set of all strategies for player i is denoted as Σ_i. A **strategy profile**, $\sigma = (\sigma_1, \ldots, \sigma_{|N|})$, is a vector of strategies (one for each player) that describes how the entire set of players jointly act. We denote the strategies in σ excluding σ_i as σ_{-i}.

Given a strategy profile, we can define several useful concepts used throughout our own work and related research in computational game theory. The **sequence probability**, $\pi^\sigma(h)$, of a given history h is the probability of reaching h if all of the players acted according to σ. Formally, $\pi^\sigma(h) = \prod_{h'a \sqsubseteq h} \sigma_{P(h')}(a|h')$, where $\sigma_i(a|h') = \sigma_i(a|I_{h'})$ for $i \in N$. This can be decomposed into each player's and chance's probability of playing to reach h. Specifically, observe that $\pi^\sigma(h) = \prod_{i \in N \cup \{c\}} \pi_i^\sigma(h)$ where $\pi_i^\sigma(h)$ denotes player i's **realization weight** (*i.e.*, probability) of taking actions to reach h under σ_i. We let $\pi_{-i}^\sigma(h)$ be the probability of everyone except player i (including chance) playing to reach h. Furthermore, let $\pi^\sigma(h,z)$ be the probability of reaching z under σ given that $h \sqsubset z$ has already occurred. Formally,

$$\pi^\sigma(h,z) = \prod_{\substack{h'a \sqsubseteq z \\ h \sqsubset h'a}} \sigma_{P(h')}(a|h').$$

The **expected utility** for player i under strategy profile σ is simply $u_i(\sigma) = u_i(\sigma_i, \sigma_{-i}) = \sum_{z \in Z} u_i(z)\pi^\sigma(z)$. This can be calculated exactly by traversing the game tree, or approximated through Monte Carlo sampling (*i.e.*, playing a large number of games).

Game theory describes several types of strategies that an agent may want to calculate or approximate. First, a **best response** for player i is any strategy that maximizes i's expected utility with respect to the strategies of the other agents, σ_{-i}. Player i's **best response value** for such a strategy is defined as $b_i(\sigma_{-i}) = \max_{\sigma'_i \in \Sigma_i} u_i(\sigma'_i, \sigma_{-i})$. When each player's strategy is a best response to the others, this is called a **Nash equilibrium**. The strategy profile σ is an **ε-Nash equilibrium** if $u_i(\sigma_i, \sigma_{-i}) + \varepsilon \geq \max_{\sigma'_i \in \Sigma_i} u_i(\sigma'_i, \sigma_{-i}) = b_i(\sigma_{-i})$ for all $i \in N$. In other words, no player can deviate unilaterally and gain more than ε.

When $N = \{1, 2\}$ and $u_1(z) = -u_2(z)$ for all $z \in Z$, the game is called **two-player zero-sum**. We will often refer to the other agents in a zero-sum game as opponents. In a **repeated game**, players repeat the game multiple times over the course of a match, typically alternating positions (*i.e.*, being a different player). In this context, when we refer to an agent's strategy σ we mean the strategy profile $(\sigma_1, \ldots, \sigma_{|N|})$ which specifies the agent's strategy in each position. In a repeated two-player zero-sum game the **exploitability** of a strategy σ, $\varepsilon_\sigma = (b_1(\sigma_2) + b_2(\sigma_1))/2$, is the amount σ would lose playing against a worst case (*i.e.*, best responding) opponent averaged over all positions. In such games, Nash equilibria have an exploitability of 0 (or ε for an ε-Nash equilibrium) and computing or approximating such a strategy is called **solving** the game.

Although frequently viewed as simple entertainment, games provide human players with low-risk environments for developing knowledge and skills. Similarly, games can be a valuable proving ground for computer agents as we pursue the development of artificial intelligence. Many games have complex properties: like imperfect information and stochastic observations. This makes them an interesting microcosm of much more complex and ill-defined "real-world" environments. Since games usually have well defined rules and scoring functions, they provide "sanitized" domains for investigating such real-world challenges. Extensive-form games can model a wide range of sequential decision making problems including classical games like chess, checkers, backgammon, Go, and poker. In the next section, we introduce the poker games (played as repeated games) used throughout this work for experimental domains.

2.2 Poker

In many classical games, players have perfect information about the game – allowing them to fully observe the state of the game. Though interesting, perfect information games like chess lack many challenging aspects of real-world problems. John von Neumann, the founder of modern game theory, is famously recounted as observing this:

> "Real life is not like that. Real life consists of bluffing, of little tactics of deception, of asking yourself what is the other man going to think I mean to do. And that is what games are about in my theory" (Bronowski 1973).

Poker, on the other hand, is a game of imperfect information. For over seventy years, the fields of game theory and artificial intelligence have used poker to investigate how agents should make decisions under such uncertainty. For example, von Neumann and Morgenstern (1944) examine poker in their seminal book on game theory. While poker played an important role during the inception of game theory, efforts to build computer poker agents using artificial intelligence techniques were fairly limited until Billings (1995) advocated for investigating poker further.

In general, poker refers to a family of stochastic imperfect information games where players play a series of games against each other with the aim of winning as much as possible from their opponents. While there are numerous poker variants, their rules share similar structure[1]. A game begins with chance dealing (typically) private cards to each player from a set of cards called the **deck**. Typically, one or more players must then make forced bets. In the poker games used in this work, these forced bets take one of two forms: either **antes** (equal bets by all players) or **blinds** (variable bets which depend on a player's position). All bets contribute to the **pot** that is paid to the winner.

The game proceeds through a number of betting rounds where players take turns wagering that their set of cards will be strongest at the end of the game. Players can **fold** (forfeiting the game), **call** (matching the last bet), or **raise** (bet in excess of the previous bet). In **limit** poker variants, bets and raises are of a fixed size and the maximum number of raises within a betting round is limited. This is in contrast to **no-limit** variants where players have a **stack** of available money and are allowed to make an unlimited number of arbitrary (integer) bets up to their stack size.

A betting round ends if every player except the last player to raise has called the outstanding bet or folded. Between betting rounds, players' hands change in some way due to chance events. One common way this happens is that chance deals some number of **community cards** which are visible and usable by all players.

Players progress through betting rounds until the game ends in one of two ways: either all but one player has folded, in which case the remaining player wins the pot, or multiple players reach the end of the final betting round and have a showdown. In a **showdown**, all remaining players reveal their cards and form the best combination, or **hand**, of cards from their private cards and any community cards. The pot is then awarded to the player with the strongest hand (or split amongst tied players). The strength of hands are determined by preset poker hand types (Wikipedia 2015b) (*e.g.*, a pair of identical rank cards loses to a flush where all 5 cards are of the same suit).

Typically poker is played as a repeated game with players changing positions after each game. Position in poker is determined relative to the **dealer button**, which is given to an arbitrary player at the start of the first game. The order of play begins with player one – the first player to the left of the dealer – moving clockwise around the table. This gives the dealer a favourable position since they observe the actions of other players prior to taking their own actions. After each game, the dealer button moves clockwise by one seat.

This work uses several variants of poker in our empirical evaluation. We begin by introducing two smaller toy poker games — Kuhn poker and Leduc hold'em — before introducing our main domain of Texas hold'em.

2.2.1 Kuhn Poker

Kuhn poker is a toy variant of poker that is small enough that a game theoretic analysis can be done by hand (Kuhn 1950). It is a two-player zero-sum poker game with a deck of three cards: jack, queen, and king. In Kuhn poker, both players initially ante \$1 and are dealt a single private card. Betting occurs in a single betting round with betting limited to at most a single bet of \$1.

In Kuhn's analysis, he showed that strategies playing certain actions from certain information sets were dominated. For example, when holding the king (the strongest hand) a player should never fold. If all such actions are eliminated, then strategies in the resulting undominated version of Kuhn poker can be parameterized with three parameters (α, β, γ) for player one, and two parameters (η, ξ) for player two. We use this undominated Kuhn poker game in some of our experiments.

[1] For the sake of simplicity and brevity, we omit many subtle details about betting in general poker games. For readers who are interested in a more exhaustive description, there are numerous books and online resources (Wikipedia 2015a).

2.2.2 Leduc Hold'em

Leduc hold'em (Southey et al. 2005) is another two-player zero-sum variant of poker which, though larger than Kuhn poker, is still small relative to common poker games played by humans. The game begins the same way as Kuhn poker: players ante \$1 and are dealt a single private card. The deck in Leduc hold'em consists of six cards with three ranks (jack, queen, and king) and two suits. Leduc has two betting rounds with betting limited to a maximum of two fixed-size bets (of \$2 and \$4 in the first and second round, respectively) per round. After the first betting round, chance deals a public community card. In a showdown, a pair of cards is the strongest hand.

2.2.3 Texas Hold'em

Texas hold'em is a popular style of poker that is often played by humans at top-level poker tournaments like the World Series of Poker. Billings (1995) advocated in particular for Texas hold'em as a research domain due to the game's simple structure and strategic complexity. Furthermore, the Annual Computer Poker Competition (ACPC) – the premier venue for computer poker research – has used variants of Texas hold'em since it began in 2006 (ACPC 2015). The availability of diverse human and computer agents makes Texas hold'em a particularly compelling domain for agent modelling research.

Texas hold'em consists of four betting rounds. At the start of the first betting round, called the **preflop**, players are forced to bet blinds (described further below) and each player is dealt two private cards. Texas hold'em uses a standard 52 card deck with thirteen ranks and four suits. Before each subsequent betting round, chance deals some public community cards: three cards at the **flop**, then one on the **turn**, and one final card on the **river**. In the event of a showdown, players form the best five card hand from the seven (two private and five public) available cards.

Two-player (also known as *heads-up*) limit Texas hold'em is our primary experimental domain. With 10^{18} states and 3.19×10^{14} information sets, heads-up limit Texas hold'em is one of the smallest poker games that humans play competitively. It is also the longest-running event in the Annual Computer Poker Competition (ACPC 2015). We also present competition results from events at the Annual Computer Poker Competition including heads-up no-limit Texas hold'em and three player limit Texas hold'em. The blind structure for each of these games varies slightly, but each game uses the same structure as the games currently used by the ACPC.

In each variant, players that face blinds must bet either a **small blind** or a **big blind** depending on their position. Typically, in poker games with more than two players, the first player (to the left of the dealer) pays the small blind, the second player pays the big blind, and subsequent players do not face blinds. In the first round, the player following the big blind acts first. In subsequent rounds, the first player following the dealer acts first. When the game is **heads-up** with only two players, "reverse blinds" are commonly used. In this case, the first player pays the big blind and the dealer pays the small blind. Note that this means the dealer acts first in the first round. Our limit games use \$5/\$10 blinds. Within each betting round, no more than four fixed-sized bets – of \$10 in the first two rounds and \$20 in the last two rounds – are allowed (with the big blind counting as one of the four allowed preflop bets). The no-limit game we use has \$50/\$100 blinds with 200 big blind (*i.e.*, \$20,000) stacks. We use a no-limit variant called **Doyle's game** where at the start of each game, players' stacks are reset to the original 200 big blinds.

Generally, our empirical results show performance in terms of expected utility in these games. To remove the impact of the magnitude of blinds and bets in these games, values are shown in **milli big blinds per game** (mbb/g), *i.e.* one-thousandth of a big blind per game. When playing games with antes, we overload this notation slightly to mean milli antes per game (as this can be viewed as all players facing equal big blinds).

In two-player zero-sum domains, the Nash equilibrium solution concept can provide some useful guarantees. Using techniques for approximating a Nash equilibrium in an extensive-form game has become a common approach to create agents in computer poker research. These techniques, which we describe next, have improved substantially in recent years in large part due to research efforts in computer poker. However, as we illustrate in this work, Nash equilibria do not produce ideal agent behaviour as they play defensively and do not attempt to maximize utility by identifying and exploiting opponent weaknesses over the series of games. Section 2.4 describes existing techniques that extend some of these Nash equilibrium approximation algorithms to compute responses that attempt to exploit other agents while retaining some of the safety afforded by a Nash equilibrium. All of these techniques provide the core of how we will construct strategies throughout this work.

2.3 Nash Equilibrium Approximation

For two-player zero-sum games, playing according to a Nash equilibrium provides a guarantee about your expected utility against a worst case (*i.e.*, best responding) opponent. An agent playing according to a Nash equilibrium strategy in a two-player zero-sum repeated game would never lose (on expectation). In this sense, a Nash equilibrium is *optimal* and computing a Nash equilibrium is typically referred to as *solving*[2] the game. Next, we introduce some techniques for computing a Nash equilibrium in two-player zero-sum games with perfect recall.

2.3.1 Sequence Form and Linear Programming

Prior work by Koller, Megiddo, and von Stengel (1994) showed that two-player zero-sum perfect recall extensive-form games could be solved in polynomial time with linear programming using a sequence-form representation of the game. In **sequence-form**, player i's strategy, σ_i, is encoded using a **realization plan**, $\beta_i \in B_i \subseteq \mathbb{R}^{\sum_{I \in \mathcal{I}_i} |A(I)|}$, that stores the strategy's realization weights. Specifically, for each information set action pair (I, a) where $I \in \mathcal{I}_i$ and $a \in A(I)$,

$$\beta_i(I, a) = \pi_i^\sigma(ha).$$

Note that this is for any $h \in I$ since, due to the perfect recall assumption, $\pi_i^\sigma(h) = \pi_i^\sigma(h')$ for all $I \in \mathcal{I}_i$ and $h, h' \in I$.

To encode a valid strategy, realization plans must also satisfy some constraints. To describe the constraints, we first define player i's **parent** of a history h, $\rho_i(h)$, as the information set action pair of player i's last decision in h. Formally, $\rho_i(h) = (I, a)$ such that $\exists h' \in I$ where $h'a$ is the largest prefix of h where $P(h') = i$. If player i has not yet acted in h, then we let $\rho_i(h) = \emptyset$. Observe that under perfect recall, $\rho_i(h) = \rho_i(h')$ for $I \in \mathcal{I}_i, h, h' \in I$. For simplicity, we define $\rho_i(I)$ as $\rho_i(h \in I)$ for all $I \in \mathcal{I}_i$. Then a realization plan β_i must satisfy three constraints to encode a valid strategy:

$$\beta_i(I, a) \geq 0, \qquad \text{(non-negative probabilities)}$$

$$\sum_{a \in A(I)} \beta_i(I, a) = \beta_i(\rho_i(I)) \text{ for } I \in \mathcal{I}_i, \text{ and} \qquad \text{(children sum to the parent)}$$

$$\beta_i(\emptyset) = 1. \qquad \text{(root has probability 1)}$$

Note that all of these constraints are *linear* in the variables of the realization plan. If β_i satisfies these constraints, then a strategy σ_i can be recovered from the realization plan by

[2] Although solution concepts aside from a Nash equilibrium exist, we do not examine them in this work. Furthermore, technically, a Nash equilibrium only weakly solves a game. A strong solution would require perfect play even after a player makes a mistake.

normalizing the realization weight for a given (I, a) pair by the realization weight of the parent of I. That is, we let

$$\sigma_i(a|I) = \frac{\beta_i(I,a)}{\beta_i(\rho_i(I))} = \frac{\beta_i(I,a)}{\sum_{a' \in A(I)} \beta_i(I,a')}.$$

For a two-player zero-sum game, the Nash equilibrium profile (β_1, β_2) must be a solution to

$$\max_{\beta_1 \in B_1} \min_{\beta_2 \in B_2} u_1(\beta_1, \beta_2)$$

while satisfying the linear constraints on β_i. Representing β_1 and β_2 as vectors we can rewrite this as

$$\max_{\beta_1 \in B_1} \min_{\beta_2 \in B_2} \beta_1^\top \mathbf{U}_1 \beta_2$$

where

$$\mathbf{U}_i((I_1,a_1),(I_2,a_2)) = \sum_{\substack{z \in Z \\ \rho_1(z)=(I_1,a_1) \\ \rho_2(z)=(I_2,a_2)}} u_i(z)\pi_c(z)$$

is the **payoff matrix** for player i. A payoff matrix has $\sum_{I \in \mathcal{I}_1} |A(I)|$ rows, $\sum_{I \in \mathcal{I}_2} |A(I)|$ columns, and at most $|Z|$ non-zero entries. By taking the dual of the inner minimization we create a linear program whose solution is a Nash equilibrium to the game.

Using sparse matrix representations (for both the constraints and the payoff matrix), the size of this linear program is only linear in the size of the *game tree* — an exponential reduction from the previous conversion to a normal-form representation. Despite this size reduction, the memory required to solve such LPs makes them impractical for large games. For example, with 3.19×10^{14} information sets, heads-up limit Texas hold'em is currently infeasible to solve using LPs. However, LPs can be used to solve smaller games (*e.g.*, Kuhn poker, Leduc hold'em, and Rhode Island hold'em (Gilpin and Sandholm 2007)), or abstractions of larger games.

Billings and colleagues' (2003) initial application of LPs to solve abstractions of heads-up limit Texas hold'em showed promise for Nash equilibrium techniques in the domain. Although the size of games feasible with LPs has improved (*e.g.*, Pays' (2014) LP approach to heads-up limit Texas hold'em), alternative techniques for solving large two-player zero-sum extensive-form games were also developed in an effort to overcome the memory requirements for solving games using LPs. New techniques including Hoda and colleagues' (2010) Excessive Gap Technique, and Zinkevich and colleagues' Counterfactual Regret Minimization, presented next, are capable of computing Nash equilibrium approximations for games several orders of magnitude larger than LP techniques.

2.3.2 Counterfactual Regret Minimization

Zinkevich and colleagues' (2008) **Counterfactual Regret Minimization**, or **CFR**, is a state-of-the-art algorithm for approximating Nash equilibria in large two-player zero-sum perfect recall extensive-form games. It resembles an iterative self-play algorithm that simulates repeated games between the players. On each iteration t, the players evaluate their current strategy σ_i^t by computing **regrets** from simulated play that quantify how well the player could have done if it had played differently. Players then update their current strategy based on the regrets accumulated up to the current iteration. The counterfactual reasoning used in computing these regrets gives rise to the algorithm's name.

More precisely, on the first iteration of the algorithm, each player i knows nothing about the game and their strategy for the current iteration σ_i^t is initialized to an arbitrary strategy. On each iteration t, players evaluate σ_i^t relative to σ_{-i}^t. At each information set, $I \in \mathcal{I}_i$, player i computes its expected value under σ^t assuming that they play to reach information

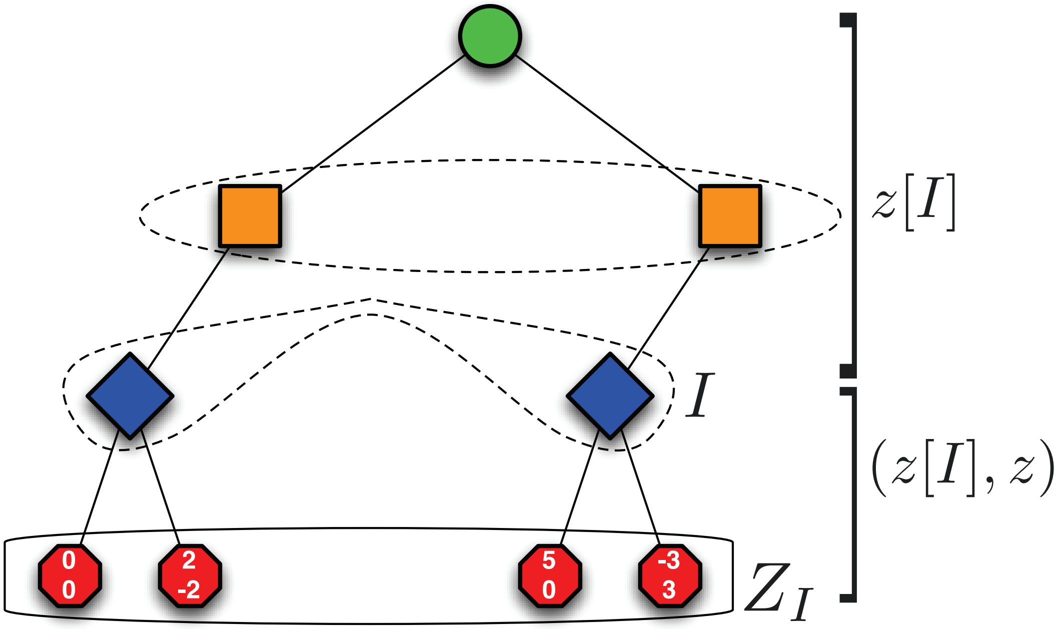


Figure 2.2: An illustration of the terms used in computing counterfactual values. In this example, we consider player two's information set where player one chose to keep the mystery box.

set I. Hence, this expected value is defined as the **counterfactual value** $v_i(\sigma, I)$ for player i. We introduce two terms prior to defining this formally. First, let $z[I]$ be the prefix of terminal history z that is contained in I. Second, let Z_I be the subset of terminal histories that pass through information set I. Figure 2.2 illustrates these terms using the Mystery box game (fully shown in Figure 2.1). Then the counterfactual value is defined as

$$v_i(\sigma, I) = \sum_{z \in Z_I} \overbrace{\pi^{\sigma}_{-i}(z[I])}^{\substack{\text{Opponents} \\ \text{reach } z[I]}} \underbrace{\pi^{\sigma}(z[I], z)}_{\substack{\text{Players reach} \\ z \text{ from } z[I]}} u_i(z).$$

CFR computes counterfactual values using a recursive walk of the game tree. As the recursive walk returns these values up the tree, CFR uses them to compute counterfactual regrets. The **counterfactual regret**, $r_i^t(I, a)$, represents how much player i wishes they had played – or how much they *regret* not playing – action a as opposed to σ^t, assuming they played to reach I. Formally,

$$r_i^t(I, a) = v_i(\sigma^t_{(I \to a)}, I) - v_i(\sigma^t, I),$$

where $\sigma_{(I \to a)}$ is the profile σ except at I action a is always taken. On each iteration, counterfactual regrets are accumulated for each information set action pair and players update σ^t to take actions proportionally to the actions' positive regret. It is important to note that, in practice, the current strategy σ^t does not need to be stored explicitly since the accumulated counterfactual regrets implicitly define it.

Updating in this fashion is referred to as **regret matching** (Hart and Mas-Colell 2000; Zinkevich et al. 2008). As a regret minimizer, regret matching ensures that the average counterfactual regret at each information set approaches zero over time. As shown by Zinkevich and colleagues (2008, Theorem 3), minimizing the counterfactual regrets at each information set in turn minimizes the **average overall regret**:

$$R_i^T = \max_{\sigma' \in \Sigma_i} \frac{1}{T} \sum_{t=1}^{T} \left(u_i(\sigma', \sigma_{-i}^t) - u_i(\sigma_i^t, \sigma_{-i}^t) \right).$$

Finally, a well-known folk theorem states that, in a two-player zero-sum perfect recall game, if both players' average overall regret is bounded by ε then the (sequence-form) average $\bar{\sigma}^T$ of the current strategy profiles σ^t is a 2ε-Nash equilibrium (Zinkevich et al. 2008, Theorem 2). Therefore, CFR will converge to a Nash equilibrium as the number of iterations increases. Although this convergence is only guaranteed in the two-player, zero-sum, perfect recall setting, CFR has also been successfully applied to multiplayer games (Abou Risk and Szafron 2010), non-zero-sum games (Johanson et al. 2011), and imperfect recall games (Johanson et al. 2013).

Computing exact counterfactual values requires a potentially expensive full traversal of the game tree. Many CFR variants mitigate this problem by estimating the counterfactual values using some form of Monte Carlo sampling (Zinkevich et al. 2008; Lanctot et al. 2009; Johanson et al. 2012a).

CFR is memory efficient, requiring memory only linear in the game's *information sets*. This has enabled CFR variants to produce approximate Nash equilibria in considerably larger games than previous LP approaches. However, human-scale games like heads-up limit Texas hold'em have historically been too large to solve without a lossy abstraction of the game. Recently though, a new CFR variant called CFR$^+$ (Tammelin et al. 2015) has been able to produce an approximate Nash equilibrium for heads-up limit Texas hold'em that (essentially) solves the game (Bowling et al. 2015). This groundbreaking result means that approximating a Nash equilibrium in heads-up limit Texas hold'em is now feasible without abstraction. That said, abstraction, which we discuss in Section 2.5, remains important both in this work and whenever available computational resources are inadequate for the domain at hand.

Although Nash equilibrium solutions provide a valuable worst-case performance guarantee, the assumption that our opponent is a worst-case adversary is not ideal when the other agents do not conform to this pessimistic assumption. While using abstraction may mean the opponent is only the worst case opponent within an abstracted version of the game, a Nash equilibrium still acts defensively: guarding itself against an adversary it may not actually be faced with. In the next section, we introduce techniques for constructing strategies that attempt to exploit some knowledge of the other agents' behaviour while remaining robust to some degree against worst-case adversaries.

2.4 Robust Responses

In two-player games Nash equilibria provide a useful worst-case performance guarantee. However, agents rarely behave in such a worst-case manner and Nash equilibria do not attempt to capitalize on opportunities, such as opponent errors, to improve their performance. An ideal agent should maximize its performance with respect to the *actual* agents it interacts with.

In an effort to achieve this ideal, one may wish to construct a response strategy that makes less pessimistic assumptions about how the other agents act. For example, we could assume a variety of models for how other agents act including: worst-case in some abstraction of the game, a model of bounded rationality, or according to some knowledge gained from prior interactions (*e.g.*, observations). Although making less pessimistic assumptions can improve our agent's performance, responding to such beliefs is risky as our agent may need to deviate from "safer" strategies, like a Nash equilibrium, to do so. In adversarial settings, this deviation exposes the agent to potential counter-exploitation by a savvy opponent who is itself modelling the agent's behaviour.

Consider the situation where we are playing against an agent who is exploitable and we know their strategy (*e.g.*, always play rock in rock-paper-scissors). Employing a best response strategy to this known opponent (always play paper) would, by definition, maximally exploit them. However, this is only the case if this opponent model is accurate. If the model is not accurate for the opponent at hand, then the best response strategy can perform poorly and be vulnerable to counter-exploitation (always play scissors) (Johanson, Zinkevich, and Bowling 2008). Model error may be caused by several factors including incorrectly modelling the parameters of the opponent's strategy, or estimating model parameters from noisy observations rather than knowing the opponent's exact strategy. In complex domains – where model error is particularly likely due to the challenge of estimating the parameters of a high dimensional strategy – robustness to such counter-exploitation is especially important.

Robust responses offer a compromise between the brittle optimism of best responses and the pessimism of Nash equilibria: trading off maximally exploiting an agent model with minimizing one's own exploitability. Next, we introduce several robust response techniques used throughout this work as we strive to produce ideal agent behaviour.

2.4.1 ε-safe Responses

McCracken and Bowling (2004) pointed out a potentially false assumption made in agent modelling literature: that an agent can, in fact, be modelled by the modelling agent. Failing to model an agent correctly due to making false assumptions about how they will act – for instance, that they are static, or that they use a particular abstraction – can cause many techniques for exploiting that model to catastrophically fail (*e.g.*, best responses). To address this issue, they introduce the concept of ε-safe strategies: the strategies that risk no more than ε compared to the value guaranteed by a safety strategy such as a Nash equilibrium. ε-safety confers exploitive strategies with a degree of robustness by bounding the maximum loss that they can suffer in cases where agent modelling has failed.

McCracken and Bowling's Safe Policy Selection (SPS) algorithm uses ε-safe strategies to exploit agents in repeated games. Specifically, SPS acts according to an **ε-safe best response**: any ε-safe strategy that maximizes utility under the current agent model. On each iteration of the game, SPS computes its strategy, acts, updates the model from observations, and then adjusts ε (increasing it over time, and increasing or decreasing it based on received utilities). This allows an SPS agent to dynamically adjust its "aggressiveness" in an attempt to maximally exploit while providing a guarantee on average rewards in the worst case.

ε-safe best responses can be computed online in small games using linear programming. Unfortunately, this approach is not feasible for larger games like Texas hold'em due to both memory constraints and the need to act online in real time. However, the restricted Nash response technique provides an alternative approach for computing ε-safe best responses that can be scaled to such complex domains.

2.4.2 Restricted Nash Responses

Johanson and colleagues' (2008) **Restricted Nash Response (RNR)** technique provides an efficient way to compute ε-safe best responses to a known adversary strategy σ_{fix} in large two-player zero-sum extensive-form games. The RNR technique computes a Nash equilibrium for a modified game using any game solving algorithm, such as an LP, or more scalable techniques like CFR.

In the modified game, the opponent is "restricted" by chance (privately) forcing them at the start of each game to play according to σ_{fix} with probability p, and with probability $(1-p)$ they are free to choose their actions. We call these two parts of the restricted player's strategy their **model** and **response**, respectively. The other "unrestricted" player is always free to choose their actions. In the Nash equilibrium of this modified game, the unrestricted player's strategy is an ε-safe best response to σ_{fix}, where ε is controlled by the choice of p. Note that solving this game only provides a strategy for one position in the game, and producing a restricted Nash response strategy for all positions requires solving a modified game from each player's point of view.

By changing p, a spectrum of Pareto-optimal ε-safe responses can be generated that trade off the response's own exploitability and its exploitation of σ_{fix}. Increasing p results in responses that are both more exploitive and more exploitable: with $p = 0$ producing an unexploitable Nash equilibrium and $p = 1$ yielding a best response. Note that although best responses are technically ε-safe for sufficiently high values of ε, we are typically interested in ε-safe best responses that risk losing an acceptably small amount.

Empirical results demonstrated that RNRs could sacrifice only a small amount of utility relative to a true best response when playing against σ_{fix} while vastly reducing the response's own exploitability (Johanson, Zinkevich, and Bowling 2008, Figure 1). Although RNRs provide substantially more robust responses than best responses, Johanson and Bowling (2009) showed that RNRs (and by extension best responses) can behave poorly when σ_{fix} is estimated from observations of the opponent rather than the opponent's true strategy. While RNRs are well suited for building responses to known strategies, data biased responses were created to address the more practical problem of building robust responses to agent models constructed from observations.

2.4.3 Data Biased Responses

In practice, it is atypical to have an agent's full strategy available and practitioners will want to build responses to some model of the agent constructed from observations of the agent's actions. For instance, one could construct a **frequentist model** of the agent by counting the number of times the agent took each action in each (potentially abstract) information set. A frequentist model could be turned into a strategy for the agent by normalizing the frequency counts at each information set and imputing some behaviour at unobserved information sets.

RNRs could be used to build a robust response to such a strategy. However, frequentist models estimate the agent's strategy with varying accuracy depending on how frequently each information set is observed. For example, in poker the information set for the first player to act is always observed (provided cards are revealed in a showdown), whereas players and chance must take specific actions to reach later information sets. Since RNRs only use a single parameter to control how much weight is put on the opponent acting like the model, they lack the capacity to represent variable confidence in the model. As a result, RNRs tend to put too much faith in the model at information sets computed from limited observations, or in the imputed behaviour.

Johanson and Bowling (2009) examined RNR's "overfitting" problem and introduced **data biased response (DBR)** strategies as an alternative way to produce robust responses. As with the RNR algorithm, DBRs solve a modified game in which a player is restricted to play according to an agent model. In DBRs however, the agent is forced *at each information set I* to act according to a frequentist model with probability $P_{\text{conf}}(I)$, and is free to best respond with probability $(1 - P_{\text{conf}}(I))$, where the confidence level P_{conf} is a function of the number of times n_I that I was observed. Since the confidence is information set dependent, DBRs avoid trusting the model in situations with sparse or non-existent data and instead fall back to assuming a best responding adversary (as opposed to some behaviour imputed a priori). Although this modified game does not admit a linear programming formulation, it can be solved with CFR techniques. Solving this modified game not only produces a robust response to the data, it also yields **mimic** strategies for the restricted players that attempt to imitate the data while also being robust to the other players.

Practitioners can control the trade-off between exploitation and our agent's exploitability by tuning P_{conf}. Johanson and Bowling's empirical results suggested that varying $P_{\max}$ with their "0-10 linear" function, $P_{\text{conf}}(I) = P_{\max} \min(1, n_I/10)$, worked well. By varying $P_{\max}$, the maximum probability of following the agent model, a range of robust responses can be produced. Note that for $P_{\max}$ of 0, the model is ignored and we get a Nash equilibrium. Empirical results have shown that DBRs behave well under a range of $P_{\max}$ values, improving considerably over RNRs in terms of both exploitation and the response's exploitability when using observations of an agent's play.

In complex domains like Texas hold'em, some form of generalization is typically required not only to make the aforementioned strategy generation algorithms feasible on available hardware, but also to effectively capitalize on available observations when building robust responses.

2.5 Abstraction

Human-scale problems are typically too complex for modern computing hardware to solve. For those human-scale problems that are feasible, the computational resources necessary to do so are often in excess of what is available to many practitioners. For instance, essentially solving heads-up limit Texas hold'em required approximately 900 CPU-years of processing (Bowling et al. 2015). In problems of such complexity, agents may only be able to learn about a limited subset of the problem's state space. **Generalization** attempts to answer how to generalize from such limited knowledge to inform us about other, potentially unobserved, parts of the state space.

Generalization is related to the task of abstraction. Where generalization relates knowledge about a subset of a problem to the larger whole, **abstraction** examines how to simplify a problem by removing or altering details of the original problem. Both areas allow for practitioners to simplify complex problems such that they will be tractable on available hardware. Of course, simplifying the problem may yield a poor surrogate for the original problem if important information is discarded. Despite the risk of less than ideal agent behaviour, generalization and abstraction are typically necessary for such human-scale problems and are ubiquitous in both human and computer agents.

In the context of extensive-form games (or other models of games), the standard approach to make a problem tractable is to derive a simpler **abstract game** by applying state-space abstraction. Strategies can then be produced for the abstract game using standard strategy generation techniques. For instance, applying a game solving algorithm such as CFR to the abstract game approximates an **abstract game Nash equilibrium**. To act in the real game, we generalize from the abstract strategies by mapping sequences of observed real game actions into the abstract game and **translating** the abstract strategy's actions back into the real game. This process is illustrated in Figure 2.3. Note that state-space abstraction is typically used to produce an abstract game *prior* to computing a strategy. Although techniques like Waugh and colleagues' (2015) recent CFR variant, regression CFR, avoid this a priori abstraction by learning a regressor during strategy generation to provide generalization, we limit our attention to abstractions applied prior to learning.

The reduction in size necessary to produce a tractable abstract game from a human-scale game typically requires at least some loss of information. The impact this loss will have on the effectiveness of the resulting strategies depends on the size of the abstract game and the abstraction technique used to produce it. Increasing the size of the abstract game increases the time and memory required to produce a strategy for the game, but

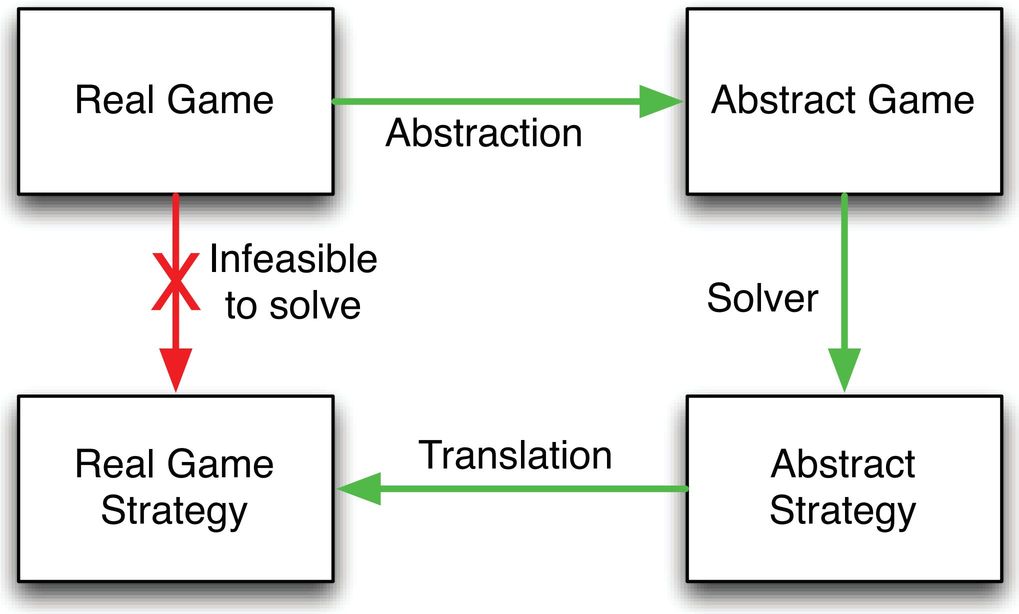


Figure 2.3: Process for simplifying and solving a game using abstraction.

also allows for less loss of information. While intuition may suggest that larger, higher fidelity, abstract games will result in abstract game Nash equilibria that are less exploitable in the real game, there is no theoretical guarantee of this property. In fact, Waugh and colleagues (2009b) provided examples of **abstraction pathologies** in Leduc hold'em where even strict refinements of an abstraction resulted in increased exploitability. In human-scale games, however, this intuition typically holds with larger abstract games yielding less exploitable strategies (Johanson et al. 2011; Johanson et al. 2013).

Prior poker research has introduced abstraction techniques that largely focus on one of two primary objectives: abstracting the cards dealt by chance's actions using **card abstractions**, and abstracting the actions of actual players through **betting abstraction**. We briefly introduce some commonly used abstraction techniques for each of these areas in the next sections.

2.5.1 Card Abstraction

In poker, chance's actions are commonly abstracted by grouping information sets with different public and private cards together into abstract **buckets**. This particular form of abstraction can be represented as a many-to-one mapping from the information sets of the real game to information sets in the abstract game. Creating a card abstraction typically involves two tasks: extracting distinguishing features to represent the cards as points in a (typically low-dimensional) feature space, and partitioning the resulting points based on some notion of similarity between them. We review several approaches used in prior poker research for each of these choices.

Card Features

As cards lack an intrinsic sense of similarity, practitioners must choose a feature space to represent the cards: ideally one that characterizes the relevant properties of a set of cards for the game at hand. In poker, these features need to help answer two fundamental questions about the cards. How "strong" is my hand given any currently revealed information? How can my strength change over the course of the game?

Although the strength of a hand can be measured in many ways, our work uses Billings and colleagues' (2002) commonly used notion of hand strength. After all the public cards have been revealed (*e.g.*, in the final round of Texas hold'em), **hand strength ($\boldsymbol{HS}$)** is the probability of a given hand winning against an opponent's hand, where ties are counted as half a win. Alternatively, one can view this as the proportion of the pot a given hand is expected to win. Given the public cards and a distribution over the opponent's available private cards, hand strength can be computed by evaluating our hand against each hand the opponent could hold. Since we do not know the true distribution over the opponent's private cards, a distribution for the opponent – usually uniform random – is typically assumed. Although some prior work refers to hand strength using such a uniform random assumption as **hand rank** (Billings 2006; Johanson 2007), this work follows the more recent trend of simply calling this hand strength. Prior to the river, while public cards have yet to be dealt, hand strength is myopic: answering how strong a hand is currently, oblivious to any future cards left to be dealt by chance. Instead, many early abstractions of Texas hold'em poker (Billings et al. 2003; Zinkevich, Bowling, and Burch 2007) used **expected hand strength ($\mathbf{E}[\boldsymbol{HS}]$)**: the expectation of hand strength over all possible rollouts of any public cards yet to be revealed[3]. This has also been called 7-card hand rank (Billings 2006; Johanson 2007).

Summarizing a hand solely by expected hand strength fails to answer the question of its **hand potential**: the tendency for a player's hand to change in strength due to future chance events. For example, a hand of 5♠6♠ may have low $\mathrm{E}[HS]$ preflop, but future community cards may turn the hand into a flush or a straight with high $\mathrm{E}[HS]$. Without differentiating hands based on their potential, an agent may conflate such high potential hands with hands that have little potential to improve. The PsOpti agents attempted to differentiate these high potential hands by reserving one bucket of their card abstraction specifically for hands with a high potential to improve (Billings et al. 2003). More recent Texas hold'em abstractions (Johanson, Zinkevich, and Bowling 2008; Zinkevich et al. 2008) capture some notion of hand potential using **expected hand strength squared ($\mathbf{E}[\boldsymbol{HS}^2]$)** to bucket hands. Since $\mathbf{Var}[X] = \mathrm{E}[X^2] - \mathrm{E}[X]^2$, the $\mathrm{E}[HS^2]$ feature provides information about both the mean and variance of a hand's distribution over hand strengths. This feature makes high potential hands appear similar to stronger "made hands" by giving extra credit to "drawing hands" that require chance to reveal further public cards before their strength is more certain. However, summary statistics alone may fail to accurately represent hand potential by discarding important information about the hand strength distribution and how it changes throughout the course of a game. Instead, modern poker abstractions typically bucket based on *histograms* of the distribution (Gilpin, Sandholm, and Sørensen 2007; Johanson et al. 2013).

These hand strength features ignore the reality that the distribution of private cards a player will take to a showdown may be non-uniform and dependent on the public actions by players and chance (such as community cards). Though convenient, assuming a uniform distribution may discard information that is potentially valuable for differentiating cards. Features such as Johanson and colleagues' (2013) **opponent cluster hand strength**, mitigate this limitation by using multiple distributions to compute hand strength features.

For an agent to act optimally in the original game, they typically require knowledge about the *public* information in addition to their own private information. The hand strength features discussed so far attempt to characterize the quality of a player's *private cards* conditioned on any known public cards. Although some information about the public cards may be indirectly captured by these features – since the known public cards affect hand

[3] Technically this distribution is not known exactly due to the private cards dealt to other players. These card removal effects are typically ignored.

strength – more specific knowledge about the public cards may be needed for an agent to distinguish important situations. To illustrate, consider the situation in Texas hold'em where players are on the river and the public community cards form the strongest possible hand. In this case, an agent should never fold with any pair of private cards since they will always tie against the opponent. However, with a hand strength of only 0.5, hands in this case may be grouped with other situations where a player should at least occasionally fold, thereby discarding valuable information about the public cards. Instead of relying solely on hand strength features, practitioners can add features of the public cards to their feature space, or directly characterize the public cards through separate **public buckets** (Waugh et al. 2009a).

Bucketing

Ideally, the chosen representation will map strategically similar cards to points (*i.e.*, feature vectors) that are in some sense similar in the feature space. Then, all that remains to create an abstraction is to choose how to partition the points such that the resulting groups (*i.e.*, buckets) are, in fact, strategically similar.

One way to partition the points is to only group together points that are exactly identical. In many poker games, Texas hold'em included, this approach can be used to exploit the fact that permuting the suits of cards does not change the relative rank of a hand. Specifically, in a standard deck of cards there are four suits: diamonds $\diamondsuit$, hearts $\heartsuit$, clubs $\clubsuit$, and spades $\spadesuit$. When evaluating a hand at a showdown, hands that only differ by a permutation of the suits are ranked the same and will tie. For example, if the public cards are 2$\clubsuit$3$\clubsuit$4$\clubsuit$7$\diamondsuit$9$\diamondsuit$ then a player with $A\heartsuit K\spadesuit$ for private cards would tie with another player with $A\spadesuit K\heartsuit$. Using these card isomorphisms, we can map all suit permutations of a set of cards to a canonical representative. Although this provides a lossless abstraction of the game (assuming best responding adversaries), the reduction in game size is fairly modest.

To reduce the size of the game further, the canonical representatives are typically partitioned using lossy abstraction techniques that relax the requirement for points to be identical. One early approach was to group points by partitioning the feature space into user defined regions (Billings et al. 2003). While this approach attempts to enforce some notion of spatial similarity between points in the feature space, it does not account for how the points are distributed in the space: possibly resulting in wasted empty regions for granular abstractions. To avoid such empty regions, **percentile buckets** specified by regions that cover an (approximately) equal quantity of points have been used (Shi and Littman 2000; Johanson, Zinkevich, and Bowling 2008; Zinkevich et al. 2008). Abstractions based on percentile buckets have been sufficient for expert-human calibre agents (Johanson 2007, Section 7.3) and are used regularly in computer poker research. However, one drawback to percentile buckets is that they do not directly optimize for – and thus need not respect – the spatial similarity between the points. To illustrate, consider a heavy-tailed distribution of points that is concentrated around one value in the feature space. In this setting, percentile bucketing may separate very similar points while grouping disparate points. Most modern poker abstractions attempt to address both of these issues by using a clustering algorithm to partition the cards directly according to some sense of spatial similarity (Gilpin, Sandholm, and Sørensen 2007; Johanson et al. 2013).

In multidimensional feature spaces, it may be necessary or valuable to bucket hands conditionally rather than from the entire (joint) feature space. This type of bucketing – called **nested bucketing** – partitions only the hands consistent with the given information. To illustrate, this could be used to partition hands in a two-dimensional feature space by splitting hands into M buckets based on one dimension, then splitting the hands within each of the M buckets into N buckets based on the second dimension, for a total of $M \times N$ buckets. For example, one could bucket based on $\mathrm{E}[HS^2]$ and $\mathrm{E}[HS]$ (Johanson 2007) or on features of both public information and hand strength (Waugh et al. 2009a). This approach is also commonly used to create a **perfect recall abstraction** by conditioning on all previously revealed information (*i.e.*, the sequence of buckets from previous rounds). Although using perfect recall abstractions provides some theoretical advantages, imperfect recall abstractions can "forget", conserving resources needed to represent past information (Waugh et al. 2009a).

With memory efficient algorithms like CFR, card abstraction alone provides sufficient reduction in size so that generating strategies for heads-up limit Texas hold'em (and even three player limit Texas hold'em) is feasible on current consumer-level hardware. However, in games like heads-up no-limit Texas hold'em where players have much larger action spaces, abstraction of the players' actions is typically necessary.

2.5.2 Betting Abstraction

Since the actions of both chance and the players contribute to the number of information sets in a game, practitioners may need to alter any of the actions – by any player or chance – to sufficiently simplify the game. The choice of exactly how to abstract the game will depend on the nature of the game, and how much of a reduction in size is needed. In games like heads-up limit Texas hold'em where the players' actions only contribute slightly to the size of the game relative to chance, betting abstraction has limited potential to reduce the game size. However, no-limit poker variants may be dramatically larger, with players' variable sized bets contributing substantially, if not primarily, to the size of the game. To illustrate, heads-up limit Texas hold'em has 3.19×10^{14} information sets whereas the variant of heads-up no-limit Texas hold'em used in the Annual Computer Poker Competition since 2010 has 6.38×10^{161} despite chance's action space being the same in both games (Johanson 2013). Constructing a feasible abstraction of such games typically requires some kind of betting abstraction.

In contrast to typical card abstraction techniques, which *bucket* information sets together, current betting abstraction techniques examine ways to simplify the game by *removing* player actions entirely. Early heads-up limit Texas hold'em poker abstractions reduced the maximum number of bets a player could make within a betting round from four to three (Billings et al. 2003; Zinkevich, Bowling, and Burch 2007; Gilpin, Sandholm, and Sørensen 2007). Betting in no-limit poker games differs in that bets are not of a fixed size and the only limit on the number of bets that players can make is their available stack of chips. To abstract larger no-limit betting spaces, both the number and size of players' bets are typically constrained. Specifically, it is common to restrict bets to particular sizes that are a fraction of the current pot. These fractions are often chosen by hand (Gilpin, Sandholm, and Sørensen 2008; Schnizlein 2009), but techniques to automatically learn good bet sizes have also been explored (Hawkin 2014). Despite restricting the size of bets there may be a combinatorially large number of possible betting sequences – especially when small fractions of the pot are allowed. As in early limit Texas hold-em abstractions, this is often addressed by limiting the number of bets a player can make. However, in no-limit games, abstractions may place restrictions on each available bet size.

Translation

For an agent to act in the real game using an abstract strategy, it requires some form of mapping to translate between information sets in the real game and information sets in its abstract game. With card abstractions, the many-to-one mapping of information sets provides a natural translation. However, unlike card abstractions which coarsen the partition of information sets, betting abstractions *remove* information sets entirely. As such, a betting abstraction may not have an intrinsic representative for every sequence of bets possible in the real game, forcing practitioners to design a translation scheme. Typically, a similarity measure between betting sequences is used to translate betting sequences into the abstract game (Gilpin, Sandholm, and Sørensen 2008; Schnizlein 2009). **Hard translation** partitions the real game information sets by mapping a real game betting sequence deterministically to the most similar abstract betting sequence. The deterministic nature of hard translation makes it relatively easy to predict and exploit. To counter this, **soft translation** maps a real game betting sequence to a weighted set of abstract game betting sequences.

All of the aforementioned abstraction techniques examine the practical problem of how to simplify a problem so that a solution can be computed. Despite the diversity of techniques described, our focus was to introduce the techniques used in this work. Other approaches, like decomposing the game into smaller pieces (Burch, Johanson, and Bowling 2014), could be used to compute strategies but are not used in this work. Furthermore, the prior discussion largely ignores the important question of how to choose the size of an agent's abstraction and how that choice impacts the agent's performance. One of the contributions of this work is an exploration of that question, so we defer further discussion of it until Chapter 4. The remaining sections of this chapter introduce some of the tools that will be used in the rest of the thesis.

2.6 Multi-armed Bandits

In a multi-armed bandit problem an agent, the "gambler", is faced with K "one-armed bandits" (*i.e.*, slot machines) that have different unknown distributions of payouts. Ideally, the gambler is aiming to maximize their total reward over a sequence of plays. Because the bandits generate payoffs stochastically and (typically) only the chosen bandit's payoff is observed, the gambler faces a common trade-off between exploiting their model of the bandits and building a better model by exploring. There is a wide literature on multi-armed bandit problems that explores a variety of different assumptions about the bandits (for a survey see Bubeck and Cesa-Bianchi 2012). We briefly introduce some well known algorithms for two general bandit settings: stochastic and adversarial (non-stochastic) bandits. Multi-armed bandit algorithms are typically evaluated based on their ability to minimize some notion of regret: capturing how suboptimal the algorithm is relative to an alternative strategy. For their respective bandit problems, the algorithms we discuss provide finite time guarantees (as opposed to guarantees in the limit) on the expected regret the agent would suffer relative to the best single action in expectation. That is, the expected utility of the algorithm's selected actions will perform nearly as well as the best single action in hindsight.

2.6.1 Stochastic Bandits

In a stochastic multi-armed bandit setting, the bandits' reward distributions are assumed to be independent and identically distributed (i.i.d.), that is, that the probability of a given reward is identical on each time step and independent from the past. In multiagent settings, where the observed rewards are dependent on the behaviour of other agents that may be learning themselves, it is unlikely that this i.i.d. assumption holds. However, in poker it is currently relatively common for computer agents to use fixed strategies: typically approximations of abstract game Nash equilibria. When all players act according to fixed strategies, the reward distribution is i.i.d. and stochastic bandit algorithms would apply. We introduce upper confidence bound algorithms as they are examined both in prior poker research and later in this work.

Upper Confidence Bound (UCB)

Upper confidence bound (UCB) algorithms are based on the general heuristic of *optimism in the face of uncertainty.* At a high level, this heuristic assumes that any uncertainty about the utility of an action will work in the agent's favour: yielding higher than expected reward. UCB-based algorithms capture this optimism by effectively assuming that the bandits will payout according to an upper confidence bound on their expected utility, as opposed to the sample mean of the observed rewards. For example, on each time step UCB1 (Auer, Cesa-Bianchi, and Fischer 2002) selects the bandit whose upper confidence bound – computed based on the sample mean of the bandit's observed rewards and the quantity of times the bandit was selected – is greatest.

In multiagent settings, the reward distributions are unlikely to satisfy the i.i.d. assumption made in the stochastic bandit setting since agents may adapt their behaviour over time. This is especially true in zero-sum games like poker where the other players are, in fact, adversaries that are directly incentivized to exploit us. This makes the stochastic bandit model risky, since algorithms like UCB can suffer linear regret (*i.e.*, UCB's regret for its decisions does not improve over time) in this adversarial setting. As such, we shift our focus to the adversarial multi-armed bandit problem where the reward distributions on each time step can be chosen by an adversary.

2.6.2 Adversarial (Non-Stochastic) Bandits

At the opposite extreme from stochastic bandits is the adversarial, or non-stochastic, multi-armed bandit problem. In this problem, no i.i.d. assumption about the bandit's sequence of reward distributions is made. Instead, an adversary gets to choose the sequence of reward distributions and the agent's goal is to minimize regret for any possible adversary choice. This adversarial model captures the task we face when playing against savvy adversaries, like humans, that may well adapt to our agent's behaviour.

Hedge

Hedge (Auer et al. 1995) is a well known algorithm for addressing the full information variant of the adversarial multi-armed bandit problem (*i.e.*, where rewards for all bandits are observed). On each time step t of a K-armed bandit problem, Hedge selects an action i to play, receives a vector of rewards for all of the actions, and updates the total rewards so far $G_i(t)$. The probability $p_i(t)$ of choosing each action is specified by a normalized exponential (*i.e.*, softmax) function of the total rewards (shown in Equation (2.1)). Although we are not directly concerned with the full information problem in this work, Hedge serves as a building block for the partial information adversarial bandit algorithms we introduce next.

$$p_i(t) = \frac{e^{\eta G_i(t-1)}}{\sum_{j=1}^{K} e^{\eta G_j(t-1)}} \tag{2.1}$$

Exp3

Exp3, the exponential-weight algorithm for exploration and exploitation (Auer et al. 1995; Auer et al. 2002), *selects actions* according to a weighted mixture of a uniform random distribution and Hedge's normalized exponential function of the total rewards. Since we only observe the reward for the chosen action in a partial information bandit problem, a vector of rewards for all actions must be "simulated". By simulating rewards using importance sampling corrections to reweight the observed reward, Exp3 constructs unbiased estimators of each action's total reward. Many variants of Exp3, including Exp4, also rely on such unbiased estimators when simulating rewards.

Exp4

Auer and colleagues' (1995; 2002) Exp4 algorithm is a slightly modified version of Exp3 designed for combining the advice of "expert" strategies. One can view Exp4 as *selecting experts* each of whom recommends a strategy for choosing the underlying actions. Instead of choosing actions in proportion to Hedge's probability distribution, Exp4 acts according to the average of the expert strategies weighted by Hedge's distribution (plus some uniform probability to mix across all actions). After selecting an action, Exp4 observes a reward for the single action, and unbiased estimates of each expert's total reward are updated by simulating rewards through off-policy importance sampling corrections. Note that unlike Exp3, each expert that recommended the selected action with non-zero probability will receive feedback from the observed reward. While the aforementioned bandit algorithms provide a regret bound relative to the best action in expectation, Exp4's guarantees are relative to the best expert in expectation.

Exp3G

Kocsis and Szepesvári's (2005) generalized Exp3 algorithm (Exp3G) applies Exp3 to a set of experts, similarly to Exp4 but without uniform mixing over the actions, while assuming that an unspecified reward simulation procedure produces an unbiased estimate of the total reward. By leaving the reward simulation procedure unspecified, Exp3G allows for different approaches to producing an unbiased estimate and a characterization of the algorithm's regret in terms of the variance of the unbiased estimator.

Oblivious versus Adaptive Adversaries

It should be noted that the quality of guarantees provided by these algorithms depend on if the adversary we consider is **adaptive**, choosing the current reward distribution based on the agent's past behaviour, or **oblivious**. In particular, these algorithms bound the regret against any adversary with respect to the single action (or expert) with highest *expected* total reward. Against adaptive adversaries, this notion of regret is somewhat strange since the adversary may have played differently had we actually chosen a different action even on the first iteration. Although these regret bounds hold for an adaptive adversary, even a fairly benign adversary can make the bound relatively weak and uninformative (for an example, see Auer et al. 1995, Section 5).

The approach we take in poker is most similar to Exp3G with unbiased estimators of the expert strategies in our portfolio generated by off-policy importance sampling combined with the "imaginary observation" variance reduction technique introduced in the next section. We defer discussion of the potential drawbacks of this approach, both for poker and other domains, to Chapter 3.

2.7 Variance Reduction for Strategy Evaluation

In statistics, the parameters of a population are exact answers regarding the properties of the entire population of objects. In practice, such parameters are typically infeasible to compute as they require practitioners to examine each member of a potentially large population. Instead, an estimator – a statistic of a sample of data drawn from the population – can be used to produce estimates of an unknown population parameter. In general, these estimates and their error relative to the population parameter will depend on the given sample of data. As such, it is useful to characterize the quality of an estimator in terms of an error measure (*e.g.*, mean squared error). By reducing an estimator's variance and improving the quality of such estimators, variance reduction techniques enable estimators to be more sample efficient: generally requiring fewer samples of the population to produce a similar quality estimate.

When evaluating an agent's performance for a chosen domain, we are concerned with the space of possible outcomes of the interactions between the environment (*i.e.*, chance) and all of the agents acting within it. In general, the actions of each of the actors will affect the utility and likelihood of an outcome. Due to the potentially stochastic nature of these actions, an agent is commonly evaluated according to their expected utility over these outcomes. Like other statistical parameters, this can be difficult to compute exactly as it requires us to consider every possible outcome. Moreover, it requires knowledge of the actions' probabilities including the strategies of other agents. Often such information is not publicly available and an estimator for the expected value must be computed through samples. In human-scale problems, accurately estimating an agent's expected utility can be a difficult problem in its own right since it may require a prohibitively large number of samples. Furthermore, in many settings our ability to sample is limited: such as problems involving human agents, or in algorithms like the multi-armed bandit problems where we need to evaluate the agent online. We briefly introduce some variance reduction techniques used in this work to help alleviate this sampling burden. Although not used in this work, Davidson (2014) introduces a "baseline" method for efficient agent evaluation along with a more detailed survey of the topic.

2.7.1 Duplicate

Duplicate can be viewed as a specific instance of an antithetic variates technique. This approach reduces variance by computing an estimator using samples of outcomes that are negatively correlated with each other. It has regularly been used in several domains including poker, bridge, and scrabble. In Texas hold'em poker, a set of duplicate sample hands can be produced by first having chance deal a set of private cards to each of the n players, and the five public cards. Then each of the set of $n!$ duplicate hands corresponds to a permutation of the player positions. For example, in a two-player game, consider the hand where chance dealt the first player $A\heartsuit A\diamondsuit$, the second player $2\clubsuit 7\spadesuit$, and dealt $A\clubsuit 3\spadesuit 4\spadesuit 5\spadesuit 6\spadesuit$ for public cards. Then in the duplicate hand player one would be dealt $2\clubsuit 7\spadesuit$ and player two would receive $A\heartsuit A\diamondsuit$, while keeping the public cards the same. By forcing the players to play each position of a deal of cards, variance is reduced since chance's ability to deal "lucky" cards to a single player is mitigated.

Some notable limitations of duplicate are that it must be employed by the dealer (since the players cannot manipulate the cards), and players must forget the hands they observed prior to playing another side of the cards. Although these limitations are not a hindrance when performing an empirical evaluation of computer agents, duplicate is not viable for other settings of interest like estimating the expected utility for our portfolio of strategies online during a match or in head-to-head human play. We introduce variance reduction techniques applicable in these online settings next.

2.7.2 Importance Sampling

One common challenge when constructing an estimator is that samples may be drawn according to a distribution $g(X)$, called the **sampling distribution** (or **proposal distribution**), that is different from the underlying population's distribution $f(X)$. Importance sampling corrects for this by reweighting the samples by a factor of $f(X)/g(X)$, yielding an unbiased estimator of the population parameter. Furthermore, by choosing the sampling distribution g carefully, importance sampling can also be used for variance reduction. To minimize variance, we should sample according to

$$g(X) = \frac{|X|f(X)}{\int |x|f(x)dx}.$$

However it is impractical to sample from this distribution since it depends on the unknown expected value we are interested in. In practice, variance can still be reduced by biasing the sampling distribution towards outcomes which contribute more to the estimator's expected value.

In the context of evaluating agents, importance sampling allows us to evaluate **off-policy**: acting and observing outcomes based on one policy while evaluating another. For example, adversarial multi-armed bandit algorithms, like the Exp3 variants described in Section 2.6.2, reweight the samples drawn according to the normalized exponential distribution (plus some exploration) over the arms while computing unbiased estimates of each arm's utility. Using off-policy importance sampling has a caveat: any outcome in the support of the target distribution f must also be in the support of the sampling distribution g, that is $f(x) > 0 \Rightarrow g(x) > 0$. If not, the estimator can be biased due to this off-policy **strategy incompatibility**. In Exp3 and its variants, this condition is typically satisfied by adding non-zero exploration across the different actions or experts.

2.7.3 Imaginary Observations

In addition to showing how basic importance sampling can be used for off-policy strategy evaluation in extensive-form games, Bowling and colleagues (2008) also illustrate how to reduce variance in such games through synthetic "imaginary observations". Specifically, for an outcome (*i.e.*, terminal history) z, we can imagine all of the alternative outcomes that are consistent with the actions of other players that led to z. For example, an agent could imagine an alternate outcome where it acted to end the game or had different private information. In poker, this corresponds to an agent folding or making a game ending call earlier in the game, or holding different private cards.

In the full information case, where the private information of other players is known after reaching z, these imaginary observations can be used without biasing the estimator[4]. However, with partial information this guarantee does not hold in general. For instance, when using such an estimator online during a poker match, we do not observe an opponent's cards if they fold. In this case, we cannot imagine ourselves holding arbitrary private cards since the observed outcome depended on the hidden information. There is a bias-variance trade-off in using such biased estimators, but in poker Bowling and colleagues found the bias from such card removal effects was small while yielding substantial variance reduction.

Note that the potential for off-policy strategy incompatibility also extends to estimators using imaginary observations. Specifically, if we want an unbiased estimator for the target strategy σ_i when behaving according to sampling strategy $\hat{\sigma}_i$, then any terminal node that can be reached under σ_i must have a non-zero probability of being *included by the estimator* under $\hat{\sigma}_i$. Under the basic importance sampling scheme, this means that $\hat{\sigma}_i$ must play to reach any terminal node that σ_i would[5] . Formally, for all terminal histories $z \in Z$, $\pi_i^{\sigma}(z) > 0 \Rightarrow \pi_i^{\hat{\sigma}}(z) > 0$. Under imaginary observations, this inclusion constraint is not as tight since we may imagine a terminal history while actually observing other outcomes. For instance, with estimators that imagine terminal histories where i has different private information, any sequence of actions that can be realized by σ_i with non-zero probability (for some private information) must be taken by $\hat{\sigma}_i$ with non-zero probability (with some other, not necessarily the same, private information).

[4] The estimator is unbiased provided the strategies of other agents are fixed.

[5] Technically, $\hat{\sigma}_i$ need not reach terminal histories that contribute a value of zero to player i's expected utility under σ. For example, terminal histories with a utility of 0 for player i, or terminal histories that other players will not take us to.

2.8 Submodular Optimization

Submodular optimization problems examine combinatorial optimizations where the set function being optimized exhibits a diminishing returns property. Consider a set function $f : 2^V \rightarrow \mathbb{R}$ that assigns a value to each subset $S \subseteq V$ of the (finite) ground set V. For $S \subseteq V$ and $e \in V$, let $\Delta_f(e|S) \equiv f(S \cup \{e\}) - f(S)$ be the **marginal gain** or **discrete derivative** of f. Then f is **submodular** if for every $A \subseteq B \subseteq V$ and $e \in V \setminus B$,

$$\Delta_f(e|A) \geq \Delta_f(e|B). \tag{2.2}$$

Although there are other equivalent definitions for submodularity, we focus on this definition for simplicity and because it clearly formalizes the diminishing returns.

Submodular functions occur naturally for many optimization problems. There are also many connections between submodular functions and both convex and concave functions (Lovász 1983). This makes submodularity an interesting and useful property to investigate for a wide range of optimization problems. For instance, similar to convex functions, minimization of unconstrained submodular functions can be done efficiently in polynomial time. However, general unconstrained maximization of submodular functions is NP-hard. In this work, we limit our attention to maximizing submodular functions that are **monotone** (*i.e.*, for $A \subseteq B \subseteq V, f(A) \leq f(B)$) and subject to a constraint k on the cardinality of the solution. That is, we seek a solution to

$$\max_{\substack{S \subseteq V \\ |S|=k}} f(S). \tag{2.3}$$

For an unstructured f, this optimization could require searching for the proverbial "needle in a haystack". However, when f is nonnegative monotone submodular, Nemhauser and colleagues (1978) show that good approximate solutions can be efficiently computed using a simple greedy heuristic. Specifically, they show that the greedy heuristic in Algorithm 1 will obtain $(1 - 1/e)$ of the optimal solution's value in the worst case.

Algorithm 1 Greedy heuristic for submodular set functions

```
Require: Ground set $V$, a submodular set function $f : 2^V \rightarrow \mathbb{R}$, cardinality constraint $k$
  Initialization: $S \leftarrow \emptyset$
  for $i \leftarrow 1$ to $k$ do
    $e^* \leftarrow \text{argmax}_{e \in V \setminus S} \Delta_f(e|S)$
    if $\Delta_f(e^*|S) \leq 0$ then
      return $S$
    else
      $S \leftarrow S \cup \{e^*\}$
    end if
  end for
  return $S$
```

Maximum coverage and facility location problems are quintessential examples of such submodular optimizations. Before introducing those problems, it should be noted that there is considerable research on other submodular optimization problems. For readers seeking further depth on submodular maximization, Krause and Golovin (2014) provide a useful survey of the area.

2.8.1 Maximum Coverage Problems

In the basic maximum coverage problem, we are given a set of elements E, a collection of sets $R = \{r_1, \ldots, r_n\}$ where $r_j \subseteq E$, and a positive integer k. The objective of the problem is to find $R' \subseteq R$ such that it "covers" as many elements of E as possible, *i.e.* maximizes $|\bigcup_{r \in R'} r|$, subject to the constraint that $|R'| \leq k$. A simple generalization of this basic maximum coverage problem is the weighted maximum coverage problem, where elements contribute a utility of $u(e)$ to the objective when covered by any set $r \in R'$. Many facility location problems examine further generalizations of this objective. In these problems, instead of R being subsets of E that cover elements in a binary fashion, R can be viewed as a set of responses they could choose from (*e.g.*, the locations where a company holds bank accounts) and the utility obtained for a given element $u(e, r)$ also varies with the choice of response. In this framework, we seek to optimize the following objective:

$$\operatorname*{argmax}_{\substack{R' \subseteq R \\ |R'| = k}} \sum_{e \in E} \max_{r \in R'} u(e, r). \tag{2.4}$$

Unfortunately, all of these maximum coverage problems are known to be computationally hard to solve exactly. For instance, Cornuejols, Fisher, and Nemhauser (1977) highlight that when u is a nonnegative utility function, the optimization in Equation (2.4) is NP-complete through a reduction from the vertex cover problem. However, Nemhauser, Wolsey, and Fisher (1978) show that when u is constrained to nonnegative values, this objective function is monotone submodular and therefore their greedy heuristic in Algorithm 1 provides an efficient $(1 - 1/e)$-approximation to the optimal solution.

In Chapter 3, we introduce the implicit modelling framework and explain how one can use the techniques described in this chapter to build dynamic agents capable of modelling other agents despite complex domains. Furthermore, we contrast this implicit modelling approach with the typical explicit modelling approach, highlighting the benefits and drawbacks of each approach.

Chapter 3

Agent Modelling

In the most general view, agent modelling investigates techniques for learning and leveraging information about the behaviour of agents. The exact nature of information one seeks to learn for an agent model can vary substantially: ranging from low level details, like their probability of taking certain actions (*i.e.*, their strategy), to higher level knowledge, such as an agent's beliefs or goals. Practitioners may desire a model specifically for prediction or mimicry of an agent's behaviour. In this setting, optimizing the model's accuracy in terms of an error measure (*e.g.*, mean squared error on prediction accuracy) would be the primary objective. Often though, practitioners use agent modelling with the goal of finding behaviour for another agent that maximizes its utility. This is the case in multiagent domains since an agent's performance – and therefore their ideal behaviour – is contingent on the behaviour of other agents. Hence, a key capability for agents in such domains is not only to learn about other agents, but ultimately how that information can be used to respond in a way that maximizes utility. In this chapter, we present a novel characterization of agent modelling approaches that identifies two distinct approaches, discusses some of their strengths and weaknesses, and highlights examples where these approaches have been previously employed.

3.1 Explicit Modelling

Traditional agent modelling approaches observe an agent's actions and construct a generative model of their behaviour. This may involve directly estimating probabilities of actions (*e.g.*, estimating the parameters of an agent's strategy), or estimating some parameters in a more sophisticated model that indirectly describes how the agent's behaviour is generated. The high-level workflow for this approach, which we call **explicit modelling**, is shown in Figure 3.1.

Although explicit modelling can provide a direct and rich representation of other agents' behaviour, there are significant practical challenges with using this approach in complex human-scale domains. First, in complex systems, agent behaviour is likely to be complex as well, and involve a very high-dimensional parameterization. Learning such an explicit model either requires significant prior knowledge about important parameters, or a prohibitively

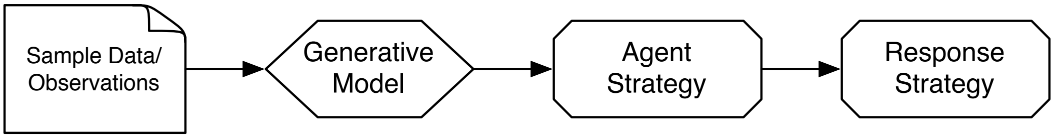
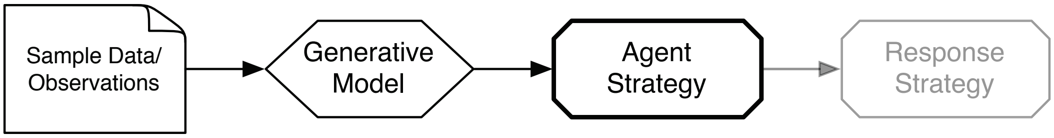


Figure 3.1: Explicit modelling workflow

large number of observations of their behaviour. In settings where we care about the utility of our agent's response to the model, we face a second challenge: how should we use the estimated model to adapt our agent's behaviour? We examine these challenges and review how prior research using explicit modelling techniques have attempted to address these issues.

3.1.1 Learning an Explicit Model



Learning a model of an agent's behaviour is itself a challenging problem. We discuss three core questions that practitioners face in learning an explicit model: How should the generative model be represented so it can be estimated efficiently? What should be assumed about an agent's behaviour when it has not been observed? If the domain has imperfect information, how should hidden information be handled?

Representation Complexity

Selecting a representation for the generative model amounts to choosing the set of parameters that will be estimated from observations and how the model space maps onto the space of agent behaviours. In order to discriminate between arbitrary stationary[1] agents in an extensive-form game, practitioners will require a generative model capable of encoding any strategy. Such expressive models are relatively easy to use in domains where the parameterization of a strategy (*i.e.*, the number of information set action pairs) is small, such as Kuhn poker (Hoehn et al. 2005; Bard and Bowling 2007) and Leduc hold'em (Southey et al. 2005). However, this approach can become problematic for complex domains like Texas hold'em due to the costs for such models in terms of memory and sample complexity.

To mitigate these costs, practitioners usually seek a parameterization for the generative model that is sufficiently compact for it to be learned and stored efficiently. Such low-dimensional parameterizations are typically based on prior knowledge of important parameters, potentially from domain experts (Southey et al. 2005), or generalization techniques. In extensive-form games, generalization allows for an observation to provide information about the agent's behaviour in information sets that are not actually observed. One intuitive way to provide generalization is to map observations from the real game into an abstract game produced with techniques like those introduced in Section 2.5. For example, Johanson and colleagues (2008; 2009) built robust responses to frequentist models that used a card abstraction of heads-up limit Texas hold'em. Similarly, Ganzfried and Sandholm (2011) used a card abstraction that discarded all private card information, but could otherwise distinguish every public state (consisting of public cards and betting). This approach provides generalization across all of the real game information sets contained within an abstract game information set. Of course, generalization can also be done in ways that do not obey the constraints of an abstract game. Vexbot (Billings et al. 2004) and BRPlayer (Schauenberg 2006) generalize across information sets using multiple schemes of varying granularity, potentially sharing information across information sets that disagree on the set of valid actions. Rubin and Watson (2012b) model an agent's "type", using only the frequency of actions in each round regardless of additional context, and indirectly map the model to an existing strategy with the most similar type.

[1] One might also wish to model dynamic agents (Bard 2008), requiring additional parameters that model how an agent's behaviour changes over time, but for simplicity we limit our exposition to choosing representations for stationary agents.

Just as abstract strategies are only able to represent a subset of the real-game strategy space, representations that discard strategy parameters sacrifice the model's discriminatory power: forcing distinct agent behaviours to be mapped to identical beliefs about the agent's strategy. In complex human-scale domains like Texas hold'em where agent behaviour may be quite sophisticated, practitioners face a trade-off between a model's discriminatory power and its sample complexity. While increasingly compact models may be easier to estimate, they also sacrifice the capacity to accurately represent the behaviour of agents we care to model. Conversely, richer models may be able to represent complex agent behaviour, but accurately estimating the model may require more observations than what is normally afforded by online interactions (especially with humans).

Unobserved Behaviour

When building a model from observations, practitioners need to handle the eventuality of having no (or limited) observations of an agent's behaviour informing the model's parameter estimates. Typically this is addressed by assuming the agent acts in some predefined way. This could consist of imputing predefined behaviour in unobserved situations (Billings et al. 2004; Schauenberg 2006; Johanson, Zinkevich, and Bowling 2008), or assuming initial parameters for the entire model. In Bayesian models, assuming initial parameters is necessary and encoded through the prior distribution. Ideally, informed priors that accurately reflect the agents being modelled would be used to improve the model's efficacy. In poker, practitioners have attempted to elicit informed priors from experts (Southey et al. 2005) and previous data (Ganzfried and Sandholm 2011). However, poorly chosen priors, like any inaccurate assumption about the agents, can have deleterious effects. For example, assuming that an agent will never act in a particular fashion may prevent the learning algorithm from discovering valuable information about the agent. As good informed priors can be hard to obtain, uninformed priors are also regularly used (Hoehn et al. 2005; Southey et al. 2005; Bard and Bowling 2007).

Practitioners have tried to mitigate their reliance on such assumptions by using an initial exploration phase to construct their model before exploiting its information (Hoehn et al. 2005; Ganzfried and Sandholm 2011; Rubin and Watson 2012b). However, using an exploration phase also raises questions of how long to perform exploration and how to act during the exploration. While Hoehn (2006) explored both of these questions in Kuhn poker, they have received little attention in human-scale domains like Texas hold'em and exploration is often done in a relatively ad hoc fashion. Note that even with an arbitrarily long initial exploration phase, a poor choice of exploration strategy can leave parts of the agent's behaviour unexplored, forcing the model to rely on some initial assumptions. For instance, Ganzfried and Sandholm (2011) found that using an (abstract) equilibrium during their initial exploration could lead to poor performance because it would not act to fully explore the other agent's behaviour.

Imperfect Information

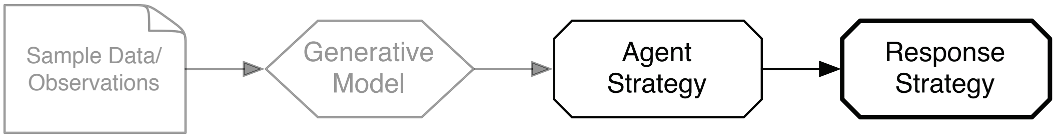
When selecting a representation for a generative model in imperfect information domains, practitioners must be careful when addressing any hidden information, such as poker's hidden private cards. Although this problem can be avoided when full information happens to be available – as in the case of hindsight logs (Johanson, Zinkevich, and Bowling 2008; Johanson and Bowling 2009; Rubin and Watson 2012a) – agents often need to adapt without such information.

Another way to avoid hidden information is to choose a representation for the model that ignores any information that may be hidden (*e.g.*, private cards in poker) and imputes a strategy that is consistent in some way with the modelled public information. For instance, Rubin and Watson (2012b) model only the frequency of an agent's public actions and map the model to an existing strategy with similar frequencies (which they construct offline using full information logs). Ganzfried and Sandholm (2011) frame the imputation as an optimization problem: solving for a strategy that is both consistent with the model's observed action probabilities and closest to a given (Nash equilibrium) strategy according to some notions of spatial distance in the strategy space. While this approach removes the need for full information, avoiding private information entirely sacrifices the model's capacity to discriminate between strategies that act distinctly based on their private information.

Ideally, an agent should capitalize on whatever information is available. One could try to gather more information by also modelling the private information in situations where it becomes public (*e.g.*, when private cards are revealed at a showdown in poker), but this can lead to other problems. For example, the observation models used in Vexbot and BRPlayer attempt to do this, however Schauenberg (2006, Chapter 4) notes that these models could represent behaviour that would not be realizable by any legal strategy. Untempered by exploration or a prior, an agent's response to such an illegal model might keep it from ever remedying the impossible beliefs.

One model that correctly accounts for the hidden information in hold'em poker games is the Bayesian model introduced by Southey and colleagues (2005). Unfortunately, the cost of exactly computing the posterior distribution over the possible strategies the agent could be using is expensive, except in small games with relatively few observations. For human-scale domains like Texas hold'em, Southey and colleagues instead sample strategies from the prior and compute a posterior over the samples. Although this approach demonstrated fast learning (Southey, Hoehn, and Holte 2009, Section 9.3), it depends heavily on the prior distribution's quality and effective dimensionality (which determines the number of samples needed to cover the prior distribution).

3.1.2 Using an Explicit Model



Learning an explicit model is only part of the agent modelling task in settings where improving an agent's *utility* is the practitioner's primary objective. In these settings, there is also the question of how an agent should act to exploit the agent model. We discuss several existing approaches to answering this question: imitation, best responses, robust responses, and model matching.

Imitation

One approach, which has the benefit of avoiding the need to compute a separate response to the model, is to assume that the model itself produces desirable behaviour and mimic the agent. This approach of learning from demonstration has been applied in a variety of settings (Argall et al. 2009). In poker, Rubin and Watson (2012a) used case based reasoning to construct models of strong agents from past Annual Computer Poker Competitions and then played according to these expert imitators. Another ACPC competitor, Marv Anderson[2], took a similar approach by imitating the play of previous ACPC winners using an artificial neural net (ACPC 2015). Though this imitation approach has proven useful, it should be noted that in poker these imitation agents were constructed offline using full information logs from prior ACPC events. This mitigates two challenges that would otherwise be faced by practitioners performing explicit agent modelling online: obtaining a large number of observations, and handling the unknown private information of other agents.

[2] Marv entered under the names of "Calamari" in 2011, "Marv" in 2013, and "Escabeche" in 2014

The fundamental drawback of imitation is the assumption that the other agent's behaviour is desirable to emulate. To highlight the distinction between being able to model an agent's strategy and being able respond effectively to it, consider the agent that always plays rock in rock-paper-scissors. Although explicitly modelling this agent would be simple to do, imitating the always-play-rock strategy will fail to produce ideal agent behaviour. To avoid this fundamental limitation of imitation, we consider other techniques that use an explicit model to produce a separate response to the observed behaviour.

Best Responses

In general, the ideal response would be one which maximizes the sum of all future expected utility given our beliefs about the other agents. However, this computation is typically intractable. A natural approach that is practical to compute is to construct a best response to the estimated model for the next iteration of the game: greedily maximizing the agent's expected utility given its current beliefs. Online agent modelling techniques that compute a best response to an explicit model have been explored several times in prior poker research. For example, Bayesian models that compute and respond to the posterior over opponent strategies have been applied in several poker games including Kuhn poker (Hoehn et al. 2005; Bard and Bowling 2007; Southey, Hoehn, and Holte 2009), Leduc hold'em (Southey et al. 2005), and Texas hold'em (Southey et al. 2005; Ganzfried and Sandholm 2011). Similarly, Vexbot (Billings et al. 2004) and BRPlayer (Schauenberg 2006), early adaptive agents for Texas hold'em, used the miximax game-tree search algorithm for computing a best response to a frequentist model.

Despite the fact that these best response based explicit modelling techniques enjoyed some success, there are several things to consider when reviewing their prior results. First, results for Vexbot and BRPlayer generally used longer matches (at least 40000 games) to learn and adapt to their opponents than typical humans could be expected to play. Second, many of the positive results were obtained against opponents that are bad (*e.g.*, always call or always raise), weak by modern standards (*e.g.*, PsOpti and Poki (Davidson 2002)), or drawn from the learning agent's own prior (Southey et al. 2005). Finally, most of the (statistically significant) analysis was done empirically with regards to opponents that were static or oblivious. Since these opponents do not directly attempt to counter-exploit the learning agent, the results provide little support for the robustness of this approach when faced with more sophisticated adaptive adversaries.

Even assuming the aforementioned challenges of learning an explicit model can be resolved, unless we are completely certain of the model's accuracy, Johanson and colleagues' (2008) results suggest that a robust response would be preferable to a best response. Figure 3.2 illustrates an example of their results and how, relative to robust responses, best responses can sacrifice considerable worst-case performance for very marginal improvements in utility against the particular agent being modelled. One might argue that such theoretical worse-case performance is unlikely to be realized since, in practice, other agents may not act in such a worst-case manner. However, even when other agents are not directly incentivized to achieve worst-case behaviour, best responses optimistically assume an accurate model of other agents' behaviour when making potentially marginal decisions. Contrary to common beliefs, this makes best responses a potentially risky approach even in non-adversarial domains such as the ad hoc teamwork problem explored by Barrett and Stone (2015). Furthermore, note that the simplest approach to compute a best response to a given model is to deterministically choose the utility maximizing action at each information set. By acting deterministically, these pure strategies remove one source of stochasticity that other agents would need to contend with in order to adapt to our response. In adversarial settings, this predictability makes the theoretical worst-case performance of best responses a much more credible threat since savvy adversaries can more easily learn and counter-exploit the best

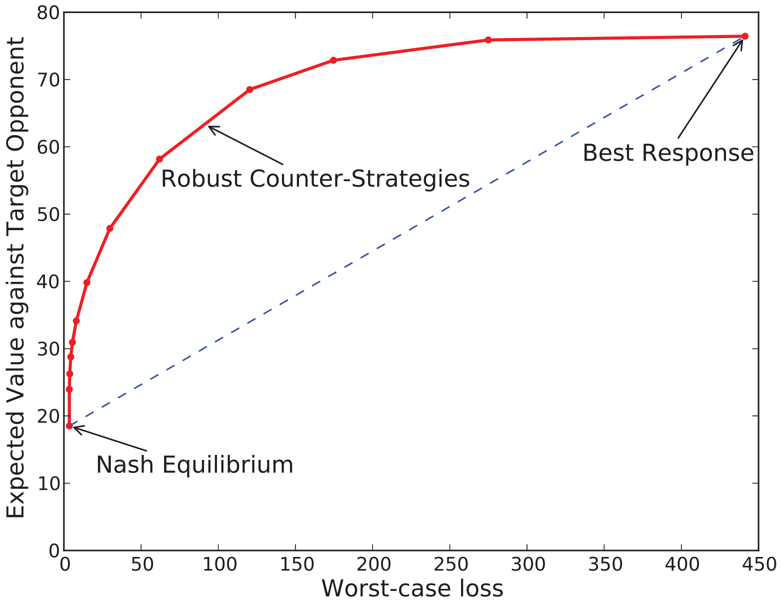


Figure 3.2: Strategies with Pareto optimal trade-off between model exploitation and worst-case performance (in an abstraction of heads-up limit Texas hold'em). Strategies were produced using Johanson and colleagues' (2008) restricted Nash response algorithm.

responder. In addition to the potential for poor worst-case performance, Johanson and colleagues' results also suggest that best responses to frequentist models generalize badly in practice compared to their robust restricted Nash responses: potentially performing poorly even against highly related opponents.

Unfortunately, having total certainty in the model's accuracy is generally impossible to guarantee. Provided that an explicit model's chosen parameterization is capable of representing the agent's behaviour, the model may be difficult to estimate for many possible reasons including: stochastic observations, imperfect information, dynamic behaviour from other agents, or having a limited time to observe the other agents. Even if the agent was *given* the current strategies of the other agents, dynamic agents (*e.g.*, humans) may adapt their behaviour over time and invalidate our existing models. Practitioners often face at least one of these challenges; in poker games we face all of them.

Robust Responses

Ultimately, if we seek to create ideal computer agents that are able to exceed human capabilities at learning and adapting to other agents in human-scale online learning problems, then those agents will necessarily need to respond despite having an inaccurate model. This suggests that it will be critical for such an agent to employ some alternative to best responses that is robust to model error.

Prior work introduced other approaches for responding to a model that do not rely on a strict best response. In early poker research, the predictability of deterministic miximax strategies was sufficiently concerning to motivate using the miximix game-tree search algorithm to produce stochastic strategies with ad hoc (Davidson 2002) or softmax (Schauenberg 2006) exploration. Rubin and Watson (2012b) respond to a model by stochastically choosing a response from the N highest utility adaptations of an existing strategy, as specified by a parameterized adaptation scheme. One can view this approach as mixing between the N best responses within a strategy space constrained by the initial strategy and chosen adaptation scheme. While these approaches may avoid true best responses – potentially making the agent's response harder to model and exploit – this does not necessarily improve the agent's worst-case performance and it is unclear how vulnerable these approaches are to adaptive adversaries.

In contrast, robust response techniques directly address an agent's trade-off between exploiting a model and worst-case performance. As Figure 3.2 illustrates, best responses and Nash equilibria (in two-player games) are at opposite extremes of this trade-off. Ganzfried and Sandholm (2012) provide another alternative in games with *gift strategies*. They present three algorithms for adaptive agents that can safely exploit an agent model: guaranteeing the agent's total expected utility over a match is at least as great as the worst-case total expected utility of an optimal safety strategy (*i.e.*, Nash equilibrium). In essence, their approach relies on accruing gifts over the match to offset the potential worst-case performance of a response to the agent model. Two of their algorithms achieve this despite using arbitrarily exploitable best responses to an agent model. Their third "risk what you've won in expectation" (RWYWE) algorithm is based on ε-safe responses and is effectively subsumed by McCracken and Bowling's (2004) more general SPS algorithm.

This approach seems appealing for human-scale domains not only due to the safety guarantee, but also because it can be achieved without needing computationally expensive ε-safe responses. However, several observations about this approach suggest it is still an unsatisfactory solution for agent modelling. First, Ganzfried and Sandholm's empirical results in Kuhn poker suggest that RWYWE fares better than their best response based algorithms. Although the authors suggest several possible explanations for this result, one overlooked explanation is the brittle nature of best responses. Even in the best case where their agent model is accurate, relative to an ε-safe response, best responses may require their algorithms to accrue considerably more gifts in exchange for marginal improvements in exploitation of the agent model. However, in the likely case that the agent model is inaccurate, the gifts may simply be squandered for no benefit. Furthermore, their experiments unconventionally assume that players' private information is revealed at the end of each game. Removing this assumption may widen the gap between techniques using ε-safe responses and best responses since this would increase model uncertainty.

More fundamentally, requiring such a strong safety guarantee potentially restricts exploitation. For example, Ganzfried and Sandholm note that safe exploitation is not possible in rock-paper-scissors, leaving the game's Nash equilibrium as the only strategy that satisfies this safety guarantee. However, rock-paper-scissors is also a common example where exploitation is vital and Nash equilibrium strategies tend to fare poorly (McCracken and Bowling 2004). Exploitation may also be further restricted when handling imperfect information as Ganzfried and Sandholm state that this would require additional pessimism in order to provide such a safety guarantee.

Alternatively, less restrictive robust responses mitigate the potentially punitive pessimism required to provide such strong safety guarantees while also avoiding the brittle nature of best responses. McCracken and Bowling's (2004) SPS algorithm is one example of such an approach. SPS provides a more relaxed notion of safety that only *bounds* the total expected utility it can lose relative to the safety strategy. As their empirical results in rock-paper-scissors demonstrate, this additional flexibility allows SPS to improve the performance of agents despite a lack of gift strategies. Similarly, as Figure 3.2 demonstrates, ε-safe strategies can dramatically improve exploitation even for relatively small values of ε.

While computing ε-safe responses using linear programming is feasible in small domains, linear programming approaches are typically impractical for the human-scale domains we are interested in, even when computed offline. In these large domains, Johanson and colleagues' (2008) restricted Nash response technique can be used to compute ε-safe responses to a given strategy, or Johanson and Bowling's (2009) data biased response technique can be used to compute robust responses to a frequentist model constructed from observations.

Unfortunately, both RNRs and DBRs are currently computationally infeasible to use online in real time for these domains. Therefore, if robust responses are to be used in human-scale domains, they currently need to be computed offline, prior to interaction and before an explicit model can be estimated for the agents at hand.

Model Matching

Although practitioners using an explicit model often generate a response to their model online, explicit models do not preclude using offline computation. One could precompute responses to a set of agent strategies offline, and act online according to the responses whose strategies "match" the behaviour being observed. Since this approach generates a portfolio of responses offline, one can view it as a hybrid between the more conventional explicit modelling approach of computing a response tailored specifically to the agents at hand and the implicit modelling approach, which we introduce in the next section.

While model matching has the benefit of enabling practitioners to use more computationally intensive techniques to produce their responses, it does not resolve the challenge of learning an explicit model. Furthermore, it also introduces two challenges similar to those faced when using implicit models. First, how should practitioners design their portfolio of responses to maximize their agent's performance? Although we examine several ways to produce a portfolio in our subsequent exposition of implicit models, we briefly introduce some approaches taken by existing model matching work. One common approach to this problem is to build responses to an explicit model based on prior knowledge, such as observations of agents interacting. This could use any of the approaches discussed earlier in this section. For instance, Barrett and Stone's (2015) ad hoc teamwork research in the RoboCup 2D soccer simulation league uses fitted Q-iteration on observations to approximate a best response for their Markov decision process model. In Bayesian frameworks the prior distribution could be used as a source of knowledge, and the portfolio could be generated by responding to models sampled from it, as in the Thompson's response technique used by Southey and colleagues (2005).

Second, how does the agent's behaviour adapt given an estimate of the generative model? In particular, it is unclear how practitioners should identify which responses in their portfolio are based on strategies that correspond to the behaviour being observed. A natural way to identify corresponding models is to compute the posterior distribution over the models (Southey et al. 2005). Unfortunately, using a model's probability of reproducing some observed behaviour to measure correspondence can be problematic. For example, if a given agent model would never produce the observed behaviour, the model will be assigned a probability of zero despite potentially being accurate otherwise. While technically correct, this is undesirable. Barrett and Stone (2015) avoid this problem by using a multi-armed bandit technique to produce a distribution over the agent models from feedbacks that depend on the probability of each arm's model reproducing an observation. Finally, spatial distances like Manhattan distance (Rubin and Watson 2012b), have also been used as measures of similarity. Implicit modelling is strongly related to model matching and can be viewed as a utilitarian matching scheme that identifies desirable responses without using a model of agent behaviour to evaluate similarity.

3.2 Implicit Modelling

In this thesis we propose implicit modelling of agents as a practical approach to adapting agent behaviour for complex human-scale domains. Rather than using observations to estimate a generative model of the other agents' behaviour, **implicit modelling** summarizes their behaviour through the (expected) *utilities* of a **portfolio** of expert strategies. This is a subtle yet key distinction between explicit and implicit models. When agent modelling

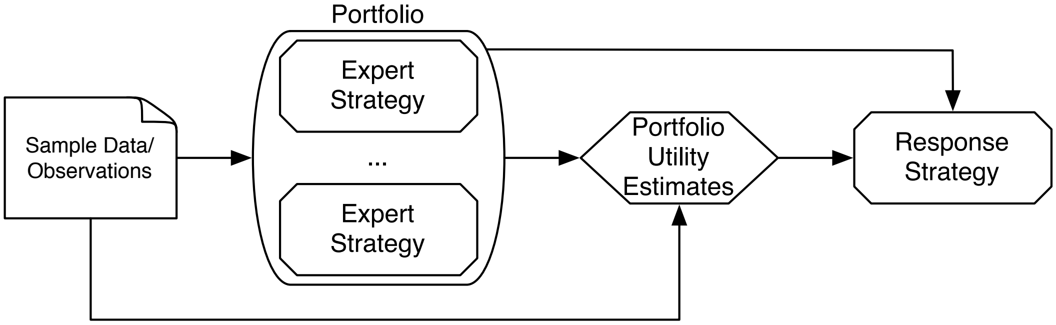


Figure 3.3: Implicit modelling workflow

practitioners take an explicit modelling approach to the problem, they use an intermediary step of estimating the other agents' behaviour in order to produce a response. In the implicit modelling approach, practitioners directly evaluate the quality of the portfolio's available responses. The portfolio's utility estimates can then be used to inform an agent on how they should combine the expert strategies to respond to the agents at hand. This process is illustrated in Figure 3.3. In the same sense as Auer and colleagues' (1995) "experts", the portfolio's strategies are unrestricted and may be stationary or dynamic, possibly changing based on prior observations. This makes the implicit modelling framework very inclusive, allowing arbitrary combinations of stationary strategies generated offline prior to interaction and dynamic strategies that change their behaviour online (*e.g.*, explicit agent models).

By avoiding the production of a generative model, implicit modelling confers several advantages over explicit modelling. First, the implicit model's representation greatly simplifies many of the challenges inherent in learning and using an explicit model. In particular, implicit models can avoid the typical "curse of dimensionality" faced in explicit modelling. For explicit models, the number of parameters generally grows with the complexity of the domain (*e.g.*, the number of information sets). In contrast, the parameterization of an implicit model is unchanged and remains the utilities for the portfolio of responses, regardless of the complexity of the domain or agent behaviour[3]. Implicit models also do not require practitioners to impute behaviour for the other agents when their behaviour is unobserved. In imperfect information domains, implicit models also avoid challenges due to hidden information since only the *utility* of an outcome needs to be observed. Finally, implicit models provide practitioners with a natural representation for using strategies that have been generated offline without the constraints imposed by real-time interaction. This is vital in complex human-scale domains where the computational cost of generating strategies, and particularly robust responses, can vastly exceed the time available to act. By affording for offline strategy generation, implicit modelling can be used in human-scale domains with relatively limited computing resources during online interaction. Of course, using an implicit modelling approach comes with its own challenges, namely, how to build an effective portfolio and how to use the portfolio online. We examine these challenges next, first answering how to dynamically adapt our strategy online using a portfolio of responses as this choice impacts the portfolio's construction.

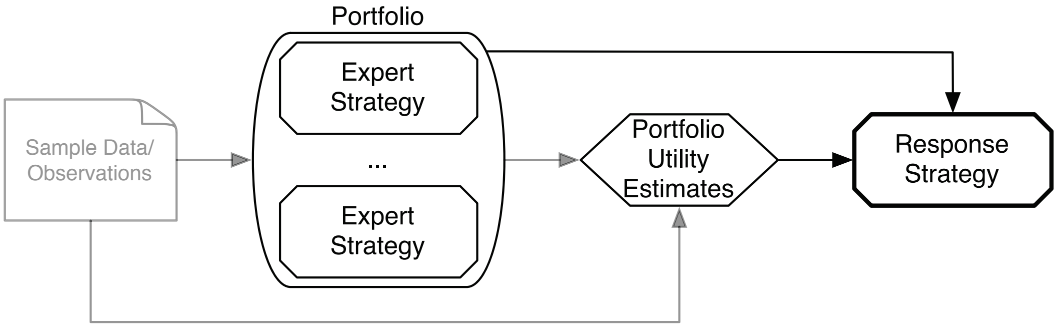
3.2.1 Using a Portfolio

If an oracle provided the expected utility for each of the portfolio's responses, we would simply act according to the response with highest expected utility to maximize our utility in

[3] Formally, in extensive-form games, one can view the implicit model as a set of linear projections of the sequence-form representation of the other agents' joint strategy. And so one can view the implicit model as a particular low-dimensional representation of the agent's strategy.

the long term. However, this information is typically unknown at the start of interaction and must be estimated online from observations. In domains where observations are stochastic, any outcomes we observe only provide noisy information about the expected utility of any particular expert. In such cases, there is a trade-off between exploiting the current utility estimates and taking suboptimal exploratory actions to refine the model's estimates. Thus, given a portfolio of expert strategies, how do we estimate the utilities of the portfolio from online interaction, and how does the agent's behaviour adapt given these estimates?

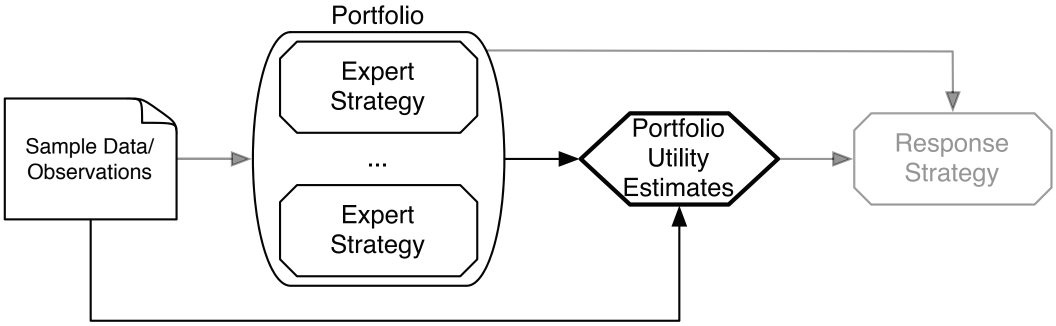
Strategy Adaptation



This general question has been investigated in a wide range of settings in reinforcement learning and multi-armed bandit literature. By using this utilitarian approach, we can leverage a variety of techniques from existing multi-armed bandit literature for this problem. For example, practitioners could view the portfolio of expert strategies as "low-level" actions in a multi-armed bandit and apply techniques like UCB1 or Exp3 to choose amongst the strategies. However, techniques that treat the experts as "meta-actions" that propose a strategy over the low-level actions, like Exp3G and Exp4, can exploit opportunities to share information from observations across multiple experts.

Unfortunately, the ideal solution depends on many factors including if the observations are generated by a stationary stochastic process, an oblivious adversary, or an adaptive adversary that conditions their behaviour on past observations. Therefore, practitioners seeking to use implicit modelling should tailor their approach to the nature of their domain, their portfolio of experts, and type of online interactions they expect.

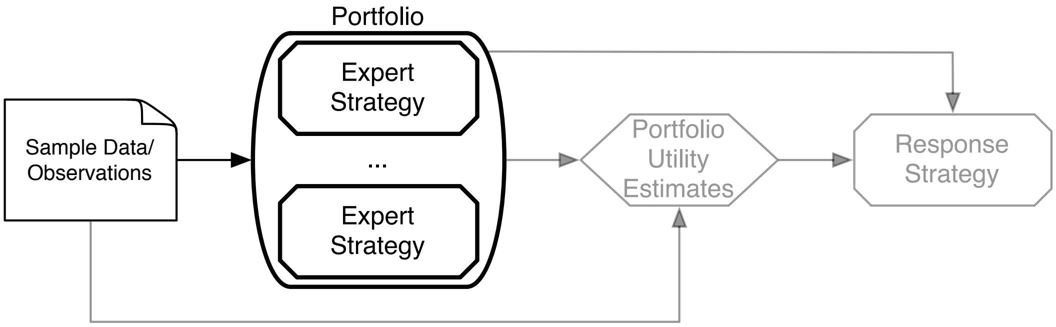
Utility Estimation



In the implicit modelling approach, the quality (*i.e.*, error) of the estimators for the portfolio's utilities directly impacts our ability to learn and adapt. For example, as Kocsis and Szepesvári (2005) show, the regret bound of Exp3G directly depends on the variance of the chosen utility estimators. For complex human-scale domains, stochastic actions, such as those of chance, can induce noise that dramatically increases the number of observations necessary to accurately estimate an agent's expected utility. Existing variance reduction techniques, like those presented in Section 2.7 or by Kocsis and Szepesvári (2005), can substantially improve the quality of such estimators and reduce the number of observations needed to respond correctly.

For our application of implicit modelling to poker, detailed in Chapter 6, we employ Bowling and colleagues' (2008) general technique for evaluating strategies in extensive-form games. Their approach combines importance sampling for off-policy estimation, akin to Exp4 or the likelihood ratio based estimates discussed by Kocsis and Szepesvári (2005), with their imaginary observations technique. This enables practitioners to use each observation of the opponents' choices and the outcome to produce low variance estimates of each response's utility simultaneously.

3.2.2 Building a Portfolio



Even with an effective algorithm for estimating a portfolio's utilities and adapting an agent's behaviour, the performance of an implicit model depends substantially on the portfolio's expert strategies. How then should practitioners design their portfolio of expert strategies to maximize their agent's performance? As multi-armed bandit algorithms may cause an agent to act according to any strategy in the portfolio – either due to noisy utility estimates or through deliberate exploratory actions – the general quality of all the strategies in the portfolio is important. As such, practitioners may seek to mitigate the potential worst-case performance of any given expert by making each of them safe to some degree. Unfortunately, current approaches for providing such worst-case guarantees in extensive-form games require a robust response or Nash equilibrium, both of which are infeasible to compute online in human-scale domains. Therefore, in such domains, practitioners must choose between a portfolio of safe experts that are all computed offline prior to interaction, or including riskier adaptive experts.

In this work we explore how to construct a portfolio of strategies offline. Not only does this allow us to produce relatively safe experts, but it also entirely removes the need to explicitly model other agents online. However, this approach comes with a challenge. Since the experts must be generated prior to interaction, the portfolio may not have an expert that is well suited to respond to the behaviour of the agents at hand σ_{-i}.

Conceptually, one could avoid this by covering the entire strategy space Σ_{-i}. Since there is a pure (*i.e.*, deterministic) strategy best response to any given agent strategy σ_{-i}, a portfolio consisting of every pure strategy would cover the space. However, this approach has two drawbacks. First, as with other best responses, these pure strategies are likely to be far more brittle than than a robust response. Second, this reintroduces the curse of dimensionality problem faced with explicit modelling since the number of pure strategies grows exponentially with the number of information sets. Although some of the pure strategies may be unnecessary, as they may not be a best response to any agent behaviour, it is unclear how to efficiently identify and remove such redundant responses. In practice, practitioners create a relatively small variety of experts with the hope of adequately covering the space of agent behaviours.

For practitioners who lack any prior knowledge about the agents they may encounter, equilibrium-based strategies are a convenient approach. For example, a variety of Nash equilibria could be used since the expected utility of any particular Nash equilibrium depends on the behaviour of the other agents[4]. For abstract game Nash equilibria, the practitioner's choice of abstraction impacts the equilibrium's performance due to the assumptions the abstraction makes about how players discriminate between information sets. We explore how the size of an abstraction impacts different performance measures of an agent's strategy in Chapter 4. Game payoffs can also be **tilted** (Johanson et al. 2011) to produce different styles of play, as in the 2008 Man-vs-Machine poker competition. While these equilibrium-based strategies do not require specific knowledge about the other agents' behaviour, they tend to make overly pessimistic assumptions about the behaviour of the other agents (*e.g.*, that they are worst-case adversaries in two-player zero-sum games).

Practitioners that have some prior knowledge about the other agents from past interactions could exploit this information by including responses to explicit models of the agents in the portfolio. Interaction, in this case, is very loosely defined. There might be data from other agents playing the game, or even complete agent strategies available. Both of these might be publicly available (*e.g.* logs from an open competition), from the designer (*e.g.* a previously available non-modelling agent), or from the agents' own past matches. While responses to explicit models could be constructed with techniques like imitation and best responses, generating a portfolio offline affords us the time necessary to build more sophisticated responses in human-scale domains. In this setting, the restricted Nash response and data biased response techniques introduced in Section 2.4 are ideal tools for generating our portfolio of responses. The choice of algorithm rests on the type of available data: the RNR algorithm works best with a complete strategy, while the DBR technique is specifically designed to handle a collection of observed games. For practitioners with knowledge about numerous agents, it may be desirable to respond to clusters of agents to limit the size of the portfolio. We introduce a decision-theoretic approach for clustering strategies in Chapter 5.

Implicit agent models have been explored several times in prior work. In robotic soccer, Bowling and colleagues (2004) used a sleeping experts bandit algorithm to select from a portfolio (or "playbook") of handmade plays to adapt their team's joint behaviour to their adversaries in the RoboCup Small-Size League. Barrett and Stone (2015) evaluate both an explicit model matching approach and an implicit modelling approach for selecting policies to cooperate with unknown teammates in the RoboCup 2D soccer simulation league. In Kuhn poker, Hoehn (2006) (from whom we adopted the explicit and implicit modelling terms) evaluated various agent modelling techniques, including implicit modelling approaches using various adversarial multi-armed bandit techniques to mix between a portfolio of pure strategy experts. In heads-up limit Texas hold'em, Johanson and colleagues (2008) evaluated an approximate Nash equilibrium and two implicit models that used UCB1 to select amongst portfolios of experts constructed using either best responses or restricted Nash responses to frequentist models. While the portfolio of RNRs outperformed the equilibrium approach against both a training and holdout set of opponents, the portfolio of best responses only fared better than the equilibrium against the previously observed training opponents and performed dismally against the holdout opponents. Rubin and Watson (2011) investigated techniques for combining strategies including an implicit modelling approach using UCB1, a portfolio of their imitation-based strategies, and the (showdown) DIVAT variance reduction technique (Billings and Kan 2006).

In this work, we build on this prior research and contribute some novel solutions to the challenges that agent modelling practitioners face when using implicit modelling. In Chapter 4 we examine how the size of abstractions used to model players impacts the performance of an agent's strategy both in practice and in the worst case. Chapter 5 explores how agent strategies should be clustered to ensure that the *responses* to the clusters provide

[4] Although Nash equilibria are interchangeable in two-player zero-sum games – yielding the same expected utility against optimal opponents – the utility of different equilibria may vary due to suboptimal agents.

high utility over the given strategies. Both of these contributions provide practitioners with techniques for constructing a diversity of strategies for a portfolio. Chapter 6 describes how we combine the techniques from Chapter 2 for strategy generation, multi-armed bandits, variance reduction, and submodular optimization to develop an end-to-end approach for applying implicit modelling in human-scale extensive-form games. Finally, in Chapter 6, we empirically demonstrate the efficacy of our implicit modelling approach in two ways: relative to other common alternatives in a detailed case study, and through results from several years of the Annual Computer Poker Competition.

Chapter 4

Asymmetric Abstractions

In large decision-making scenarios it is common to use abstraction techniques to simplify the space of solutions (*i.e.*, strategies) and to make it tractable for our reasoning algorithms. This presents agent designers with the difficult task of choosing the abstractions which encode the players' knowledge of the real game. Practitioners typically face two problems in choosing an abstraction. First, as in many domains, practitioners face a feature extraction problem where they must obtain features which provide discriminatory power between states where an agent should act differently. Prior research on abstraction in poker has focussed on this problem and we discussed several of the approaches in Section 2.5. Second, the granularity of an abstraction presents practitioners with a trade-off between computational requirements and the fidelity of the abstraction. In single-agent settings, one typically uses the finest granularity possible that is tractable given the available computation.

In multiagent settings, the situation is considerably more complicated. First, it is not only necessary to choose an abstraction for the agent of interest, but it is also necessary to choose an abstraction for the other agents in the environment. Such an abstraction choice is representing one's belief about the other agents' capabilities, knowledge, and computational capacity. In this sense, a fine-grained abstraction is not always the obvious choice. Further, given fixed computational resources, it is not obvious how to trade off using those resources to have a finer-grained agent abstraction or a finer-grained abstraction for the other agents. In particular, these abstraction choices may affect two common performance measures: one-on-one performance against other agents, and worst-case performance in the unabstracted game.

The situation in multiagent scenarios is complicated by Waugh and colleagues' (2009b) abstraction pathologies, which show that in multiagent domains there is no guarantee that refining an abstraction will result in improved worst-case performance. Despite the lack of theoretical guarantees, there is considerable evidence that finer-grained abstractions do actually improve agent decision-making. For example, in Texas hold'em poker, finer-grained abstractions have been shown to perform better in head-to-head competitions (Zinkevich et al. 2008; Johanson 2007) as well as resulting in less exploitable behaviour in the worst case (Johanson et al. 2011; Johanson et al. 2013).

However, all of this empirical evidence has been based on *symmetric* abstraction choices, where the same abstraction is used for all of the agents. There has been little empirical analysis on the effect of **asymmetric abstractions** – *i.e.* different abstractions for the agents – on the quality of the resulting behaviour. The historical use of symmetric abstractions is both a default assumption and a matter of convenience, as a symmetric abstraction only requires a game to be solved once to compute all players' strategies, while an asymmetric abstraction requires us to solve the game once for each arrangement of the players' abstractions.

As a result, there is limited guidance to a practitioner for how they should trade off abstraction granularity between the agents, especially in human-scale domains. In terms of real game exploitability, Waugh and colleagues' (2009b) abstraction pathology results examined both symmetric and asymmetric abstractions in Leduc hold'em poker and showed that nothing can be guaranteed about worst-case performance in the real game if the opponent is abstracted. Further, the recent development of a variant of the CFR algorithm called CFR-BR has made it possible to solve asymmetric abstract games where the opponent uses no abstraction (Johanson et al. 2012b). This provably converges to an abstract strategy with the lowest possible real game exploitability. While these results provide valuable insight, they do not investigate the impact of abstraction size on asymmetric abstractions when both players are abstracted.

Although real game exploitability is an important objective measure of a strategy's quality, agents typically do not face their worst-case opponent and one-on-one performance against other agents may be more relevant in practice. Johanson and colleagues' CFR-BR work highlighted this by noting that despite being optimal in terms of real game exploitability, CFR-BR strategies tended to perform worse one-on-one compared to strategies solved with CFR using a symmetric abstraction. This result suggests that there may be a trade-off between real game exploitability and one-on-one performance, but this trade-off has only been investigated in the extreme case where the opponent uses no abstraction. More interesting trade-offs may occur in other asymmetric abstractions when either our agent or the opponent's abstraction is relatively stronger. Without further investigation, agent designers seeking to use abstraction techniques are left with unanswered questions about the practical uses of asymmetric abstractions.

Agent designers face even more unanswered questions when generating robust counter-strategies to agent models constructed from limited observations (as opposed to an agent's explicit strategy). In this case, even smaller domains which could otherwise be represented exactly may require designers to assume some form of generalization to prevent model sparsity (*i.e.*, insufficient sampling of agent behaviour across the model's possible decision points). This situation arises when using the DBR algorithm described in Section 2.4.3 as it uses state-space abstraction to provide this generalization. This further encumbers designers with selecting an abstraction for the model of the opponent in addition to abstractions for the robust counter-strategy being created and the opponent's unrestricted response strategy. The prior work on robust counter-strategies (Johanson, Zinkevich, and Bowling 2008; Johanson and Bowling 2009) explored only the default case of symmetric abstractions, and the possible advantages of using asymmetric abstractions have not yet been explored. Without this investigation, designers are left with a gap in guidance on how to select abstractions that will yield the best robust counter-strategies.

We contribute the first thorough empirical exploration of asymmetric abstractions. We do this in the domain of two-player limit Texas hold'em, where others have shown the value of symmetric abstractions. We examine how the abstraction trade-off affects an agent's performance in both head-to-head competition and in terms of worst-case exploitability in the real game, which is often used as a measurement of Nash equilibrium approximation quality. Our results give the first guidance for practitioners on the abstraction trade-off and show that symmetric abstractions, while being the most common choice, may not be ideal. In addition, we explore the effect of abstraction choice when building counter-strategies from observations of other agents. In this case, there is an additional trade-off of representation capacity and sample efficiency. The impact of these abstraction choices are particularly relevant to our proposed implicit modelling framework, as practitioners using the approach must build a diverse portfolio of strategies that balance exactly these trade-offs. We begin our analysis with an examination of asymmetric abstractions in the context of computing Nash equilibrium approximations and then provide further results regarding robust counter-strategy generation.

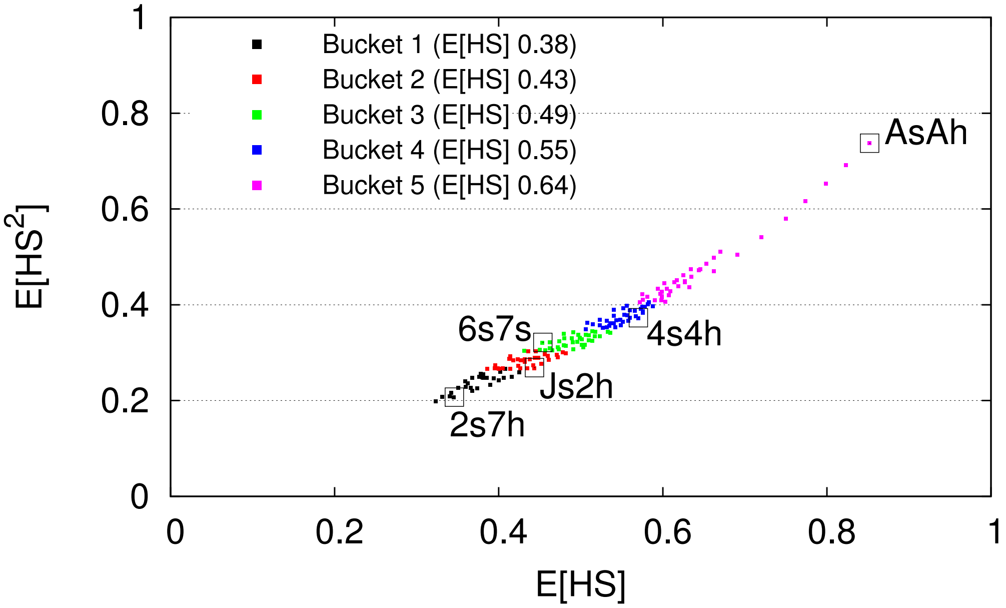


Figure 4.1: Abstraction of the first round of Texas hold'em poker, dividing 1326 hands into 5 percentile $E[HS^2]$ buckets.

4.1 Nash Equilibrium Approximation

A common approach to generating strategies in two-player zero-sum extensive-form games is to approximate a Nash equilibrium using techniques like those introduced in Section 2.3. In this setting, designers typically must choose how to distribute their limited memory between the size of the abstraction for the strategy being created and the size of abstraction for the opponent's response strategy. As discussed previously, the question of how asymmetric abstractions impact an agent's one-on-one performance and real game exploitability remains largely unaddressed. Our first experiment directly investigates this question for approximate Nash equilibria strategies.

4.1.1 Experimental Design

Throughout our asymmetric abstraction experiments, we use abstraction techniques from prior research in two-player limit Texas hold'em poker. Abstraction is applied only to the chance events in the game, and not to the players' actions. The abstraction task is thus simplified to finding similar sets of cards which are mapped together to form buckets, as introduced in Section 2.5.1. In our experiments, we use percentile buckets over the expected hand strength ($\mathrm{E}[HS]$) and expected hand strength squared ($\mathrm{E}[HS^2]$) card features. On the first round of the game, for example, a 5-bucket percentile $E[HS^2]$ abstraction groups the 1326 possible hands of cards into the buckets shown in Figure 4.1. The abstractions we investigate have the perfect recall property, in which hands that are mapped together on one betting round must also be mapped together on all earlier betting rounds. By convention, the same branching factor n is used on each round. We consider three standard sizes of abstractions from prior research, with branching factors of 5, 8 and 12 buckets on each round. The 5 and 8 bucket abstractions are partitioned according to percentile divisions of $E[HS^2]$. The 12-bucket abstraction uses a nested bucketing that first divides the hands into six $E[HS^2]$ sets, which are further split into two $E[HS]$ sets. The sizes of these abstract games and the amount of memory required by CFR to solve them is shown in Table 4.1.

Abstraction	Information Sets	CFR Memory
5-Bucket	3624290	140 MB
8-Bucket	23551424	934 MB
12-Bucket	118671936	4708 MB

Table 4.1: Sizes of percentile abstractions.

Nine approximate Nash equilibrium strategies were constructed using different pairs of abstractions for the strategy being created and the opponent's response strategy. Each player's strategy uses one of a 5, 8, or 12-bucket perfect recall percentile $E[HS^2]$ abstraction. The CFR algorithm described in Section 2.3 was used to generate a strategy for each pair of abstractions (3 symmetric and 6 asymmetric). For each of the three abstractions we also produced a strategy using the CFR-BR algorithm (Johanson et al. 2012b) which solves an asymmetric game in which the opponent uses no abstraction. Each strategy's real game exploitability was computed using the Accelerated Best Response algorithm (Johanson et al. 2011), and the one-on-one expected value, measured in milli-big-blinds per game, between each pair of strategies was measured by playing 100 million duplicate games of poker (200 million games total). The size of each abstract game is listed, with asymmetric games requiring half of each abstraction's size from Table 4.1, and CFR-BR requiring 1096 megabytes of memory for the unabstracted opponent (Johanson et al. 2012b).

4.1.2 Empirical Results

Table 4.2 presents the results for these 12 strategies. Each strategy has a label "U-R" which indicates which abstraction was used for *us* and for the opponent's *response*. For example, "12-5" is a strategy using an asymmetric abstraction where our strategy uses the 12-bucket abstraction and assumes the opponent is using a 5-bucket abstraction. Strategies labelled "U-FULL" are solved using the CFR-BR algorithm in which the opponent plays the full (*i.e.*, unabstracted) game.

These results provide insight about several trends that arise when increasing abstraction sizes. First, examining the symmetric abstraction case (5-5, 8-8, 12-12), we see that as we increase abstraction size both mean utility against the field and exploitability improve. While Waugh and colleagues' abstraction pathology results showed that this is not guaranteed by any theory, these results help explain why competitors in the Annual Computer Poker Competition (ACPC) (ACPC 2015) endeavoured to produce progressively larger abstractions (Sandholm 2010). To explore the effects of increasing each player's abstraction independently, we must move to asymmetric abstractions.

Our first significant result in asymmetric abstractions is the discovery of the first abstraction pathologies outside of Waugh and colleagues' (2009b) experiments in Leduc hold'em. Note that abstract game Nash equilibria in the 5-12 and 8-12 abstractions are both less exploitable than in the 12-12 abstraction: if our goal is to minimize our exploitability, we would do better by using a smaller abstraction for our own agent. Further, the 5-8 strategy which reduces the abstraction size for both players is also slightly less exploitable than 12-12. This counter-intuitive finding shows that abstraction pathologies are not just a problem in toy domains.

Examining the asymmetric abstractions where the opponent's abstraction is larger than our agent's, such as 5-8 or 8-FULL, we observe a trade-off between one-on-one performance and exploitability. In all cases, as the size of our opponent's abstraction increases (*i.e.*, we become more pessimistic about our adversarial opponent's abilities), our exploitability improves while our one-on-one mean utility decreases. The CFR-BR strategies (U-FULL) are the extreme case of this form of asymmetry and, as observed by Johanson and colleagues (2012b), pay a substantial cost in one-on-one utility in order to minimize ex-

	12-5	12-8	12-12	8-5	8-8	8-12	5-5	5-8	5-12	Mean	Exploitability	Size
12-5	0	-3	-6	20	18	16	43	41	41	18.970 ± 0.128	435.757	2424 MB
12-8	3	0	-3	23	22	20	36	35	35	18.890 ± 0.143	378.919	2821 MB
12-12	6	3	0	16	16	14	29	28	30	15.842 ± 0.175	289.227	4708 MB
8-5	-20	-23	-16	0	-3	2	22	21	24	0.662 ± 0.121	379.659	537 MB
8-8	-18	-22	-16	3	0	4	16	15	20	0.276 ± 0.144	312.762	934 MB
8-12	-16	-20	-14	-2	-4	0	12	12	16	-1.985 ± 0.099	255.845	2821 MB
5-5	-43	-36	-29	-22	-16	-12	0	3	7	-16.189 ± 0.112	317.1	140 MB
5-8	-41	-35	-28	-21	-15	-12	-3	0	5	-16.751 ± 0.153	283.37	537 MB
5-12	-41	-35	-30	-24	-20	-16	-7	-5	0	-19.714 ± 0.190	234.351	2424 MB
12-FULL	-22	-22	-21	-14	-13	-11	-2	-1	2	-11.526 ± 0.221	87.2765	3450 MB
8-FULL	-36	-36	-32	-26	-24	-21	-14	-12	-7	-23.093 ± 0.070	101.256	1563 MB
5-FULL	-54	-50	-45	-42	-38	-35	-29	-26	-21	-37.585 ± 0.150	122.385	1166 MB

Table 4.2: Cross table of approximate Nash equilibria strategies using various abstractions. The row player's expected value is shown in milli-big-blinds per game, rounded to the nearest integer. Values against individual opponents are computed through sampling 100 million duplicate hands (200 million hands total) and have 95% confidence intervals of at most 0.424. 95% confidence interval on the mean is shown.

ploitability. This result suggests that agent designers whose goal is to minimize worst-case performance should assume a pessimistic (*i.e.*, fine-grained) abstraction for other agents.

Finally, we investigate the asymmetric abstractions where our agent's abstraction is larger than the opponent's, such as 8-5 or 12-8. Here we see the opposite trend in the trade-off between one-one-one performance and exploitability: as we improve our agent's abstraction, our exploitability gets worse while our one-on-one performance improves not only in the mean utility, but also in the one-on-one utility against each individual strategy. This suggests that agent designers focused on the more practical one-on-one performance goal may want to increase the size of their agent's abstraction.

This investigation of Nash equilibrium approximations using asymmetric abstractions isolates two independent but related trends in how abstraction size affects agent performance. Although historically the standard approach has been to use symmetric abstractions, we have shown that this choice may be balanced but not optimal for either minimizing exploitability or maximizing one-on-one performance. In fact, these goals are at odds: the exploitability-minimizing 12-FULL strategy would lose in one-on-one play against the smallest 5-5 symmetric abstraction, while the performance-maximizing 12-5 strategy is also the most exploitable.

A designer's prior domain knowledge and their beliefs about the other agents in the environment will impact their priorities over these goals. For example, if worst-case outcomes corresponded to people being injured or killed, improving worst-case performance may be a designer's only goal. In this case, using a more pessimistic fine-grained opponent abstraction may improve worst-case performance by yielding strategies that guard against an opponent which better approximates the worst case (although abstraction pathologies prevent any guarantees). On the other hand, if a designer believes the other agents are unable or merely unlikely to act so as to produce worst-case performance, then choosing abstractions which optimize one-on-one performance, such as a fine-grained abstraction for the agent's strategy, may produce better results in practice. As we will demonstrate in the next section, these abstraction choices continue to be relevant when we create robust counter-strategies that strike a balance between these goals.

4.2 Robust Counter-Strategies

Robust counter-strategies provide another approach to constructing agent strategies. Instead of trying to optimize performance assuming a best responding opponent, as with the previous approximate Nash equilibria approach, robust counter-strategy algorithms attempt to compromise between worst-case performance and exploiting knowledge of how other agents act. When this knowledge comes from observations of an agent rather than explicit knowledge of their strategy, designers usually need to transform the observations into a generative model of the agent's behaviour. In many domains, the number of information sets may make it infeasible for the model to represent each distinct information set. Furthermore, even if every information set can be represented, a limited quantity of observations can result in insufficient sampling to build an accurate model. This **model sparsity** is typically combated by agent designers generalizing about their observations in some form. DBRs address both the model representation and observation generalization problems by using state-space abstraction.

The prior work on robust counter-strategies only examined the techniques using the small symmetric 5-bucket abstraction introduced earlier (Johanson and Bowling 2009; Johanson, Zinkevich, and Bowling 2008) and does not tease apart the distinct roles of the opponent abstractions. While the abstraction for the opponent's *response* acts to prevent the resulting counter-strategy from becoming exploitable by overfitting to the opponent model, the abstraction for the opponent *model* provides generalization across a limited set of agent observations. Without exploring asymmetric abstractions, we cannot answer how designers should select abstractions to produce the best robust counter-strategies from a given quantity of observations when using techniques based on state-space abstraction, like DBR.

Though not previously explored, note that both RNR and DBR can be used with asymmetric abstractions. In particular, recall that DBR generates robust counter-strategies using an opponent whose strategy is mixed at each information set between an unrestricted regret minimizing strategy and an opponent model (Johanson and Bowling 2009, Equation 1). These abstractions for the opponent's response and model do not need to be the same, and DBR can use a small abstraction for the model to ensure data density while using a large abstraction for the response to reduce the counter-strategy's exploitability. Likewise, the abstraction for our counter-strategy agent can be distinct from either of the opponent's two abstractions.

We will now directly investigate this aspect of creating robust counter-strategies with both symmetric and asymmetric abstractions. Though our results from Section 4.1 provide evidence for how designers should choose the abstractions for the agent's strategy and the opponent's response, an abstraction for the opponent model must also be chosen. Our experiments will directly examine how the quantity of observations and abstraction size used to build the opponent model impact DBR performance. Furthermore, we present results for RNRs and DBRs using both symmetric and asymmetric abstractions evaluated according to their true worst-case performance in the real game for the first time.

4.2.1 Experimental Design

To explore the impact of the abstraction size, we used a variety of different configuration parameters to generate several RNRs and DBRs based on knowledge of an exploitable opponent. RNRs trained using an agent's explicit strategy optimally trade between one-on-one and worst-case performance in their abstract game. In practice, it is uncommon to know an agent's strategy exactly and it must be inferred from observations. When building responses from observations, RNR is prone to an overfitting effect that DBR avoids, and so we compute DBR counter-strategies in such cases.

The opponent was created using CFR to solve a modified 8-bucket abstract game where payoffs were tilted, as in the original presentation of DBR (Johanson and Bowling 2009). Specifically, the tilted opponent (falsely) believes that it will receive 25% additional utility when it wins the pot either through a showdown or the other player folding. When building DBRs, models of the opponent were created using $10^4, 10^5, 10^6$, or 10^7 full information hands of play (*i.e.*, private cards were revealed). Observations of the tilted opponent were gathered using the same "probe" agent as in (Johanson and Bowling 2009) which never folds and calls or raises with equal probability.

Throughout these experiments we examine RNRs and DBRs generated using different combinations of the aforementioned 5, 8, and 12-bucket perfect recall percentile $E[HS^2]$ abstractions. Trend lines labelled "DBR-U-R-M" indicate that the corresponding strategies are DBRs generated using abstractions U, R, and M for the agent's robust counter-strategy, opponent's response, and opponent model, respectively. RNRs are only labelled "RNR-U-R" as their opponent model is the tilted opponent's actual 8-bucket strategy.

Finally, as described in Section 2.4 both RNR and DBR have parameters which control the counter-strategy's trade-off between one-on-one and worst-case performance. RNR uses the parameter p to specify the probability the opponent plays according to their model. DBR combines a parameter $P_{\max}$ with a function that maps a quantity of observations to a probability the opponent must play according to the opponent model. The original presentation of DBR found that the 0-10 linear function (yielding $P_{\max}$ with 10 or more observations and a linear interpolation between 0 and $P_{\max}$ from 0 to 10 observations)

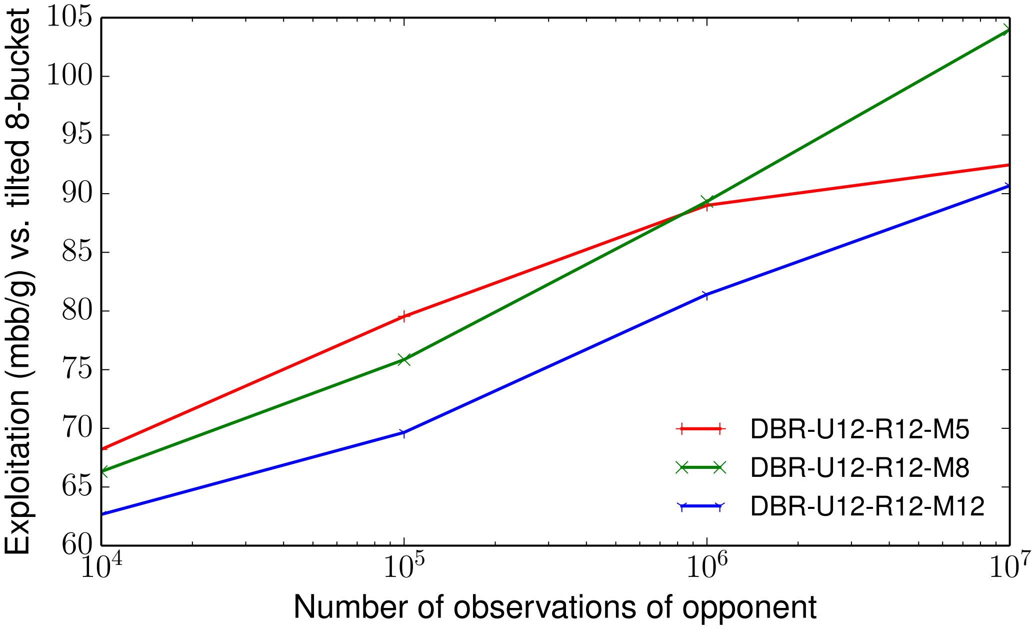


Figure 4.2: Impact of quantity of observations and model abstraction on exploitation of tilted 8-bucket equilibria. Exploitation values against the tilted opponent are in milli-big-blinds per game (mbb/g). Values were computed through sampling 100 million duplicate hands (200 million hands total) and have 95% confidence intervals of at most 0.585. Strategies for each data point are exploitable for 100 mbb/g in the 12-bucket abstract game.

performed best in practice (Johanson and Bowling 2009). We used this 0-10 linear function and varied the value of $P_{\max}$ or p to create a range of strategies.

As with our earlier Nash equilibrium results, strategies were evaluated in terms of exploitability and one-on-one expected utility against the tilted opponent. One-on-one expected utility was evaluated through sampling 100 million duplicate hands (200 million hands total). Values for one-on-one performance are in milli-big-blinds per game (mbb/g) and have 95% confidence intervals no larger than 0.585 mbb/g.

4.2.2 Empirical Results

Because DBRs are constructed from observations, the choice of abstraction for the opponent model directly impacts on our beliefs about the opponent. To isolate the interplay between the opponent model's abstraction and the quantity of observations, in our first experiment we fix the abstraction for the DBR strategy and the opponent's response strategy to the 12-bucket abstraction while varying the opponent model's abstraction size. Furthermore, we control each DBR's worst-case performance by computing DBRs that are exploitable for 100 milli-big-blinds per game in the 12-bucket abstract game. Figure 4.2 shows the one-on-one performance trends for DBRs using each of the three abstractions sizes for the opponent model as we vary the quantity of observations used to create the model.

These results demonstrate that using asymmetric abstractions for the opponent model and the opponent's response can produce strictly better robust counter-strategies. With fewer observations (10^4 and 10^5) we would improve our one-on-one performance against the tilted opponent while keeping the robust counter-strategy's worst-case performance in the abstract game the same. As the number of observations increases the DBRs using the 8-bucket opponent model eventually catch up to (10^6) and then surpass (10^7) the DBRs using the 5-bucket opponent model. This crossover point is likely due to the small number of observations being too sparsely spread across the 8-bucket opponent model. Despite incorrectly representing the tilted opponent's true abstraction (8-buckets), the coarser 5-bucket abstraction yields better generalization and a more functional opponent model than the 8-bucket abstraction with very few observations. With a larger number of observations, the 8-bucket opponent model can better populate and capitalize on its correctly chosen model whereas the 5-bucket abstraction remains an incorrect model of the tilted opponent. We also observe that the 12-bucket abstraction does poorly throughout. This is unsurprising as it would needlessly separate observations from the tilted opponent's underlying 8-bucket strategy into the larger 12-bucket opponent model, resulting in both greater model sparsity and an incorrect representation of the opponent's abstraction. This suggests that designers should aim to select opponent model abstractions that are not only similar to their opponent's true abstraction, but can also be estimated accurately from their limited observations.

Prior work on robust counter-strategies evaluated the RNR and DBR techniques in terms of worst-case performance in the abstract game. Computing the real game exploitability of strategies has only recently become feasible (Johanson et al. 2011), and as a result we can now revisit these techniques and evaluate their counter-strategies' true exploitability for the first time.

Our results in Section 4.1 suggested that increasing the size of our agent's and the opponent's response abstraction both traded between one-on-one and worst-case performance in contrary ways. Using RNRs and DBRs in different abstractions, we investigate if these trends also hold for robust counter-strategies. Performance curves were generated by computing a counter-strategy for each of the following p and $P_{\max}$ parameters: 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99, 1.0. We begin by examining the performance of RNRs in Figure 4.3 as this eliminates the impact of limited observations.

Examining the best values for one-on-one and worst-case performance, we see similar trends to those observed with approximate Nash equilibria. Comparing RNRs using a 12-bucket instead of an 8-bucket abstraction for the agent (RNR-U8-R8 to RNR-U12-R8, and RNR-U8-R12 to RNR-U12-R12), we see that the curves move up and to the right corresponding to improved one-on-one and poorer worst-case performance. Interestingly though, the RNRs using the larger 12-bucket opponent abstractions not only improve in worst-case performance relative to the RNRs using the smaller 8-bucket opponent abstractions (RNR-U8-R8 to RNR-U8-R12, and RNR-U12-R8 to RNR-U12-R12), but their performance curves also *dominate* them.

Note that since RNRs use the opponent's exact strategy, at p of 1 they become a best response to the opponent model and are oblivious to the abstraction chosen for the opponent's response. On the other hand, DBRs with a $P_{\max}$ of 1 only *approach* a best response in the limit of observations. Therefore, the opponent's response abstraction will always have some, though diminishing, impact on DBRs even with $P_{\max}$ set to 1. Figure 4.4 illustrates this difference, showing that DBRs, depending on the quantity of observations used, behave between the RNR domination and the more direct trade-off between one-on-one and worst-case performance observed with Nash equilibrium approximation. Comparing the DBRs using 10^7 observations (DBR-U12-R8-M8-10m and DBR-U12-R12-M8-10m), we see tends similar to the RNRs in Figure 4.3: using a larger opponent response abstraction produces nearly dominant performance curves. Unlike RNRs though, we can observe that using the larger 12-bucket opponent response abstraction results in a slight loss in the best one-on-one performance possible. For DBRs using only 10^4 observations (DBR-U12-R8-M8-10k and DBR-U12-R12-M8-10k) a more distinct trade-off between one-on-one and worst-case performance is apparent. With fewer observations, the DBRs using the larger 12-bucket opponent response abstraction have performance curves which shift distinctly down and to the left, corresponding to improved exploitability and poorer one-on-one performance.

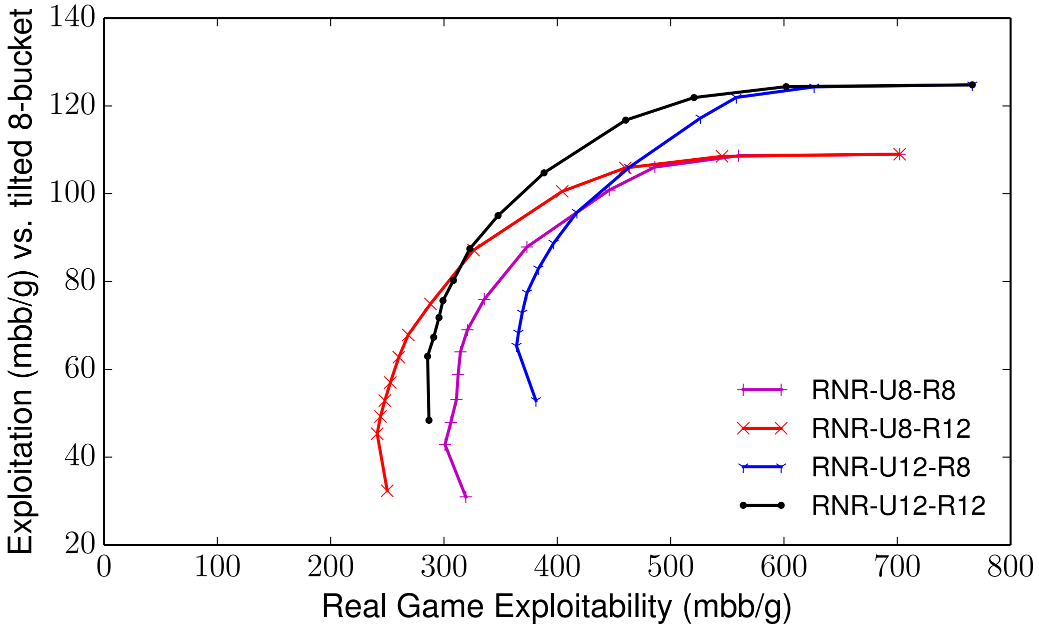


Figure 4.3: Impact of counter-strategy and opponent response abstraction size on RNR one-on-one and worst-case performance in the unabstracted game. Values are in milli-big-blinds per game (mbb/g). Exploitation values against the tilted opponent were computed through sampling 100 million duplicate hands (200 million hands total) and have 95% confidence intervals of at most 0.585.

Both the RNR and DBR results echo the findings of Section 4.1: larger abstractions for our agent's abstraction should be used for greatest possible one-on-one performance, while larger opponent abstractions can produce better worst-case exploitability.

Finally, Figure 4.5 shows how varying the opponent model's abstraction affects one-on-one and worst-case performance in the real game. We use the same abstractions as in Figure 4.2, but create DBRs trained with 10^5 observations. RNRs using the 12-bucket abstraction for the robust counter-strategy and the opponent's response are also shown. Performance curves were generated by computing a counter-strategy for each of the following p and $P_{\max}$ parameters: 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99, 1.0.

Surprisingly, we observe that the robust counter-strategies with positive weight on the opponent model can be *less exploitable* than abstract game Nash equilibria (*i.e.*, RNR or DBR with p or $P_{\max}$ of 0) using the same abstraction. Figure 4.3 demonstrates this improved exploitability with RNRs while Figures 4.4 and 4.5 show this improvement is even more dramatic for DBRs. This may allow us to construct robust counter-strategies at no cost to worst-case performance relative to an abstract game Nash equilibrium. That said, these results may be specific to the particular tilted opponent we used. Furthermore, these robust counter-strategies still do not, and could not, have better exploitability than the 12-FULL CFR-BR strategy from Table 4.2. Although the cause of this effect is currently unknown, it appears similar to an effect discovered by Johanson and colleagues (2011, Table 3) in their investigation of tiled equilibria, in which over-aggressive abstract strategies were found to be less exploitable than Nash equilibria within the same abstraction.

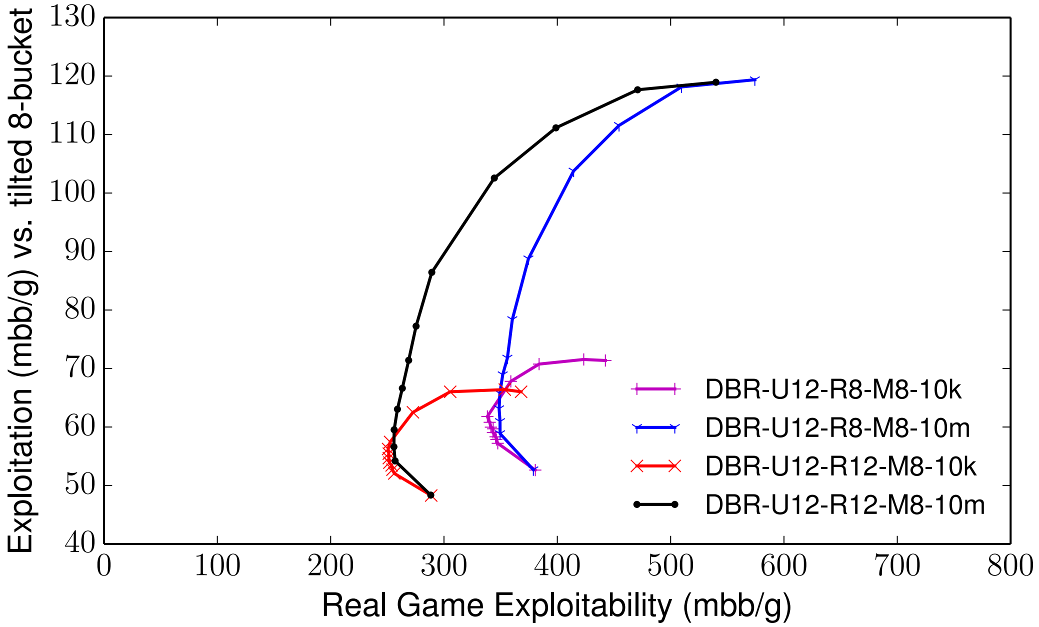


Figure 4.4: Impact of opponent response abstraction size and quantity of observations on DBR one-on-one and worst-case performance in the unabstracted game. Values are in milli-big-blinds per game (mbb/g). Exploitation values against the tilted opponent were computed through sampling 100 million duplicate hands (200 million hands total) and have 95% confidence intervals of at most 0.585.

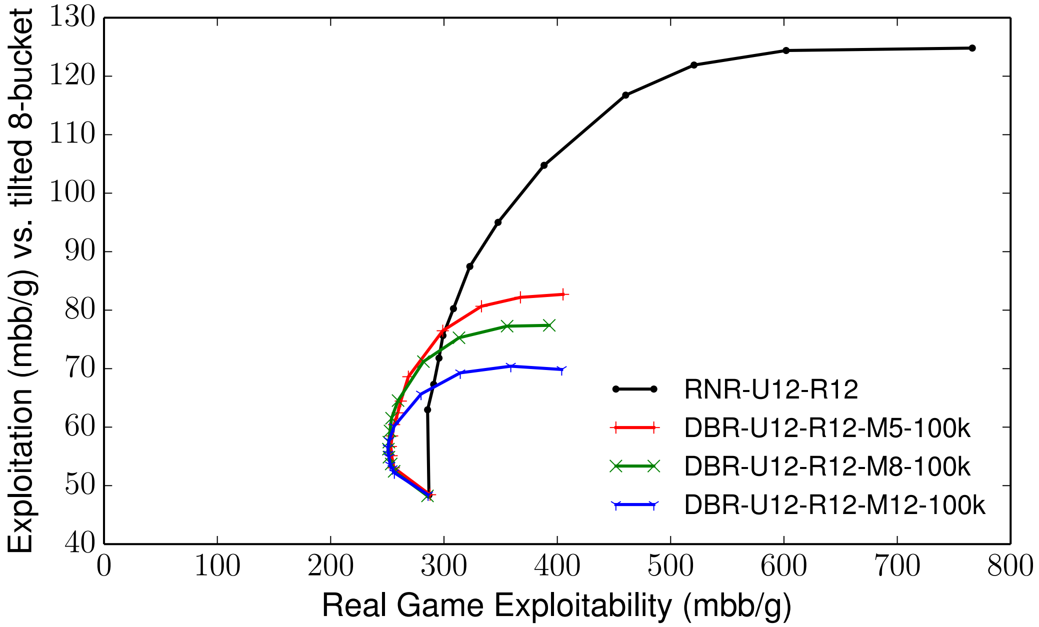


Figure 4.5: Impact of opponent model abstraction size on one-on-one and worst-case performance in the unabstracted game with 100,000 observations. Values are in milli-big-blinds per game (mbb/g). Exploitation values against the tilted opponent were computed through sampling 100 million duplicate hands (200 million hands total) and have 95% confidence intervals of at most 0.585.

4.3 Summary

In large multiagent domains where some form of state-space abstraction is necessary to make the problem tractable, agent designers are often faced with the difficult task of dividing limited computational resources for representing each agent's behaviour between the agents in the environment. While the standard approach in the poker domain is to use a symmetric abstraction that divides the resources evenly between the players, we have shown that this choice does not optimize for either worst-case performance or performance against suboptimal opponents. Our experiments performed the first empirical analysis of asymmetric abstractions in the human-scale game of two-player limit Texas hold'em, and addressed both Nash equilibria and counter-strategies. Our results demonstrate that the trade-off between between real game exploitability and one-on-one performance, which Johanson and colleagues' (2012b) first observed in their CFR-BR experiments, also exists when both players are abstracted. In the equilibrium approximation setting, we discovered the first abstraction pathologies outside of a toy domain. In the counter-strategy setting, we found that in addition to choosing abstractions for both players, the size of the abstraction used to model the opponent should be chosen to match the quantity of data. Finally, we performed the first experiments with robust counter-strategies that measured real game exploitability, and found that robust counter-strategies can occasionally be less exploitable than abstract game Nash equilibrium strategies.

These results provide valuable insights about the impact of abstraction choices which practitioners can use to guide their portfolio construction. When creating abstract game Nash equilibria, choosing asymmetric abstractions provides the portfolio with a diversity of responses that may better model the behaviour of agents with relatively superior or inferior knowledge. For practitioners seeking to generate a portfolio with responses based on prior knowledge of other agents, asymmetric abstractions may play a critical role in avoiding model sparsity when constructing those models from observations. Unfortunately, practitioners still face the question of which prior knowledge should be used to build such responses.

Chapter 5

Decision-Theoretic Clustering

In domains where maximizing an agent's utility is the primary goal, agents may need to model other agents or entities in their environment to improve their utility. Although an agent could tailor a response to each individual agent or entity it has prior knowledge of, such individual personalization is often impractical. Clustering techniques can be beneficial in these settings by allowing the agent to partition the set of entities into similar groups called clusters. In this setting, a clustering's *actionability* — its capacity to suggest responses with high utility to the agent — is the fundamental clustering objective. Contrast this with many traditional clustering problems, such as k-means and k-medians, where similarity is measured by some notion of *spatial* distance between the entities within the same cluster. For instance, the k-means objective is to minimize the within-cluster sum of squared Euclidean distances between each of the entities and their cluster's centroid. Despite an abundance of spatial clustering techniques, these techniques may fail to capture similarity in how the agent should respond to the entities.

To illustrate, we briefly examine the game of rock-paper-scissors. In this game, a static agent's behaviour can be specified by their probability distribution over choosing rock, paper, and scissors. Figure 5.1 depicts a simplex representing the space of possible probability distributions over these three actions. The point E is the game's Nash equilibrium of 1/3 for each action. Consider the points labelled 1, 2, and 3. Although 1 and 2 are spatially close, an agent's **response** for how to act with respect to them should be different as their strategies do not share the same best response. In contrast, 1 and 3 share the same best response (always play paper) despite being spatially distant.

Although this type of decision-theoretic clustering may appear to be a niche problem at first glance, it is actually related to a range of optimization problems. Kleinberg and colleagues (1998) introduced and formalized this style of clustering problem in the data mining community as *segmentation problems*. Their work showed that optimal solutions to this type of clustering induce optimal solutions to a form of maximum coverage optimization, and vice versa. Lu and Boutilier (2011) highlighted several parallels between their budgeted social choice model and segmentation problems. Carlin and Zilberstein (2008) also use a similar type of utility-based clustering to group observations in Dec-POMDPs, reducing the size of agent policies. Their algorithm, though, was presented with no approximation bounds.[1]

Due to the computational complexity of segmentation problems (and many other similar maximum coverage based problems) one typically forgoes exact solutions for efficient approximate solutions to these problems. In particular, Nemhauser and colleagues' (1978) greedy algorithm for maximizing monotone submodular functions is often used to attack

[1] The theoretical results from this chapter can also be applied to their problem, giving an algorithm with approximation guarantees for the single-agent case. Whether the same principles can give an algorithm with approximation bounds for the multi-agent case is open.

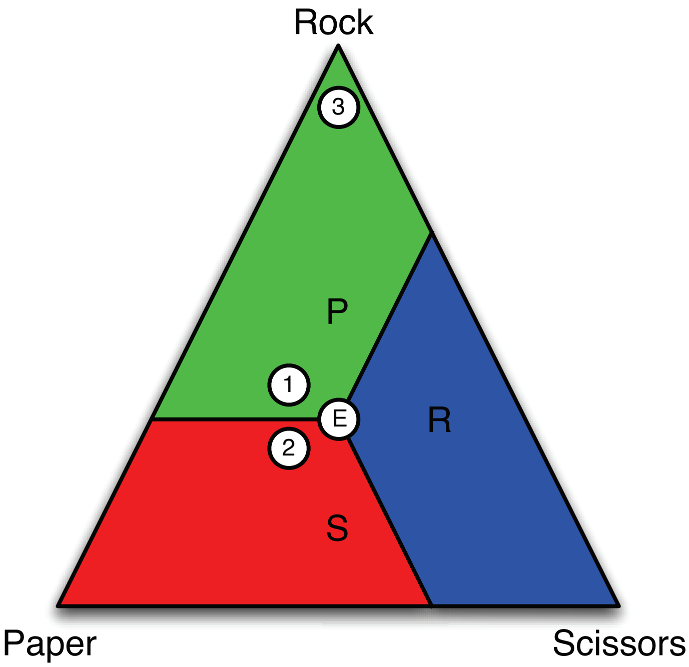


Figure 5.1: Rock-paper-scissors strategy simplex partitioned into best response regions. Despite their spatial proximity, 1 and 2 fall into distinct best response regions (P and S).

a segmentation problem's maximum coverage formulation. This approach, which we introduced in Algorithm 1 of Section 2.8, iteratively constructs a solution by evaluating the marginal gain of each feasible response, adding the greedy-best response. Unfortunately, as Kleinberg and colleagues (2004) noted, such greedy algorithms may not be efficient as even a single step can be an NP-complete problem.

In this chapter, we examine segmentation problems where the response space is either exponential or infinite and the standard greedy approximation is computationally infeasible. This scenario arises naturally when designing an implicit modelling agent's portfolio, since practitioners would seek responses that efficiently cover the space of behaviours that they expect other agents to employ. We show that despite the complexity of the response space, in certain cases where the utility function can be factored, an efficient response oracle can be constructed. We then use such a response oracle to operate directly on the partitioning formulation of a segmentation problem by greedily merging clusters together in an agglomerative (*i.e.*, "bottom up") clustering algorithm. Finally, we evaluate our technique both theoretically and empirically. Our theoretical results provide a guarantee on the worst-case performance of our greedy algorithm relative to the optimal clustering into k sets. This approximation bound is shown to be tight within a factor of 2. We empirically evaluate our technique using extensive-form games by clustering agent strategies in two toy games of poker: Kuhn poker and Leduc hold'em. Our results highlight the benefit of clustering strategies based on their actionability rather than their spatial similarity by contrasting our greedy method with a traditional k-means clustering approach. We begin our exposition by reviewing segmentation problems and introducing related notation.

5.1 Segmentation Problems

Segmentation problems (Kleinberg, Papadimitriou, and Raghavan 1998) examine the challenge of determining how an agent should respond to maximize utility given information about different entities[2]. For example, a commercial enterprise with information about their customers could act homogeneously across the customers, but the enterprise may be able to increase its utility by tailoring their **response** (*e.g.*, marketing strategy, product line) to each customer's preferences. While such individual personalization is often impractical, a more limited form of personalization where the market of customers is segmented (*i.e.*, clustered) into k groups may still be beneficial.

One way to view segmentation problems is as clustering problems where the desired clustering depends on a utility function $u(e, r)$ that specifies the utility of **response** $r \in R$ with respect to an **entity** $e \in E$ being clustered. Unlike traditional clustering approaches, this approach directly optimizes for the actionability of the clustering.

More formally, let E and R be sets (E finite), and $u : E \times R \to \mathbb{R}$ be a **utility function**. For convenience, we let $\mathrm{Part}(E)$ denote the **set of partitions of $\boldsymbol{E}$**:

$$\mathrm{Part}(E) = \left\{ P \in 2^{2^E} \,:\, \cup P = E; A, B \in P \Rightarrow A \cap B = \emptyset \right\}.$$

For $k \in \mathbb{N}$, we shall also use $\mathrm{Part}_k(E)$ to denote the **set of $\boldsymbol{k}$-element partitions of $\boldsymbol{E}$**:

$$\mathrm{Part}_k(E) = \{ P \in \mathrm{Part}(E) \,:\, |P| = k \}.$$

We abuse notation slightly to define several recurring terms for utility over subsets and partitions of E. For $P \in \mathrm{Part}(E)$, $C \subset E$, $r \in R$, let

$$\begin{aligned} u(C, r) &= \sum_{e \in C} u(e, r), \\ u(C) &= \max_{r \in R} u(C, r), \text{and} \\ u(P) &= \sum_{C \in P} u(C). \end{aligned}$$

In words, $u(C, r)$ is the total utility of a "cluster" of entities $C \subset E$ when the response is r, $u(C)$ is the utility of C and $u(P)$ is the utility of partition P.

The partitioning form of a segmentation problem considers the problem of finding a partition of E that gives the highest utility amongst all k-element partitions:

$$P_k^* \in \operatorname*{argmax}_{P \in \mathrm{Part}_k(E)} u(P) = \operatorname*{argmax}_{P \in \mathrm{Part}_k(E)} \sum_{C \in P} \max_{r \in R} \sum_{e \in C} u(e, r). \tag{5.1}$$

For clarity, note that this assumes clustering E is our ground truth objective: other potential entities are ignored. Finally, we let u_k^* denote the **utility of an optimal k-element partition**: $u_k^* = \max_{P \in \mathrm{Part}_k(E)} u(P)$.

To provide more visual intuition of this problem, we can view the problem as an optimization on a matrix. Let U be an $|E| \times |R|$ matrix where the (i, j)-th entry $U_{i,j} = u(e_i, r_j)$. Then, given $k < |E|$, the goal of the optimization is to find a partition $P = \{C_1, \ldots, C_k\}$ of the rows of U that maximizes the sum of the utilities over the rows when all rows in the same cluster must share the same response column r_j. Figure 5.2 depicts an example of this matrix form.

Kleinberg and colleagues (1998) observe that this partitioning view of a segmentation problem is also equivalent to the following weighted maximum coverage optimization which we introduced earlier,

$$\operatorname*{argmax}_{\substack{R' \subseteq R \\ |R'| = k}} \sum_{e \in E} \max_{r \in R'} u(e, r). \tag{5.2}$$

In our matrix view of this problem, one can think of Equation (5.2) as choosing the set of k columns of the matrix U that maximally cover the rows of the matrix.

[2] Kleinberg and colleagues referred to the entities and responses as customers and decisions, respectively.

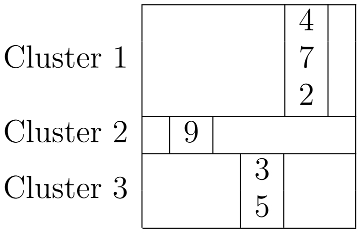


Figure 5.2: Matrix view of a segmentation problem with an objective value of 30.

The utility of optimal solutions to these two views of a segmentation problem not only have equal value, u_k^*, but one can construct an optimal solution to Equation (5.2) given a solution to Equation (5.1), and vice versa. Though straightforward, we show these constructions and their computational cost in Lemmas 1 and 2. Furthermore, we provide a proof of their equivalence in Lemma 3, showing that optimal solutions to one view of a segmentation problem induce optimal solutions to the alternate view.

Lemma 1. *Partitions induce covers.*
Given $P \in \mathrm{Part}_k(E)$ where $v = u(P)$, one can construct $R' \subseteq R$ with $|R'| \leq k$ such that $\sum_{e \in E} \max_{r \in R'} u(e,r) \geq v$ in $O(|E||R|)$ evaluations of u.

Proof. We prove this by constructing $R' \subseteq R$. Begin with $R' = \emptyset$. Then for each $C \in P$, add $r_C \in \mathrm{argmax}_{r \in R}\, u(C,r)$ to R'. This adds at most k unique elements to R'. Finally, observe that R' has a maximum coverage value of at least v.

$$v = \sum_{C \in P} \max_{r \in R} \sum_{e \in C} u(e,r) = \sum_{C \in P} \max_{r \in R'} \sum_{e \in C} u(e,r) \leq \sum_{C \in P} \sum_{e \in C} \max_{r \in R'} u(e,r) = \sum_{e \in E} \max_{r \in R'} u(e,r)$$

□

Lemma 2. *Covers induce partitions.*
Given $R' = \{r_1, \ldots, r_k\}, R' \subseteq R$ such that $v = \sum_{e \in E} \max_{r \in R'} u(e,r)$, one can construct $P \in \mathrm{Part}(E)$ where $|P| \leq k$ and $u(P) \geq v$ in $O(k|E|)$ evaluations of u.

Proof. We prove this by constructing such a partition as follows. Initialize $P = \{C_1, \ldots, C_k\}$ where $C_i = \emptyset$. For each $e \in E$, add e to C_i where $i \in \mathrm{argmax}_{j \in \{1,\ldots,k\}}\, u(e, r_j)$. Finally, remove C_i if it remained empty. P is clearly a partition of E of size at most k where $u(P) \geq v$ since for each $C_i \in P$, $u(C_i) \geq u(C_i, r_i)$. □

Lemma 3. *Mutual optimality.*

$$\max_{\substack{P = \mathrm{Part}(E) \\ |P| \leq k}} \sum_{C \in P} \max_{r \in R} \sum_{e \in C} u(e,r) = \max_{\substack{R' \subseteq R \\ |R'| \leq k}} \sum_{e \in E} \max_{r \in R'} u(e,r)$$

Proof. Suppose that P is an optimal partition of E with value v, and $R' \subseteq R$ is an optimal maximum coverage set with value $v' > v$, where $|P|$ and $|R'|$ are both at most k. Then, by Lemma 2, we can construct a partition P' of E no larger than k with value at least v' from R'. This contradicts the original optimality of P. An equivalent argument can be made for the reverse direction using Lemma 1.

□

The equivalence of these two problems means that techniques for solving the maximum coverage based problem in Equation (5.2) can also be used to solve the partitioning problem in Equation (5.1). Unfortunately, as discussed in Section 2.8.1, this type of weighted maximum coverage problem is known to be computationally hard. When u is a nonnegative utility function, the weighted maximum coverage optimization in Equation (5.2) is NP-complete (Cornuejols, Fisher, and Nemhauser 1977). Additionally, Kleinberg and colleagues (1998) show that even for several constrained response spaces (with u a general linear function), the segmentation problems are still NP-complete, or more specifically MAXSNP-complete (Kleinberg, Papadimitriou, and Raghavan 2004).

Due to these negative complexity results, approximate solutions to segmentation problems are typically sought in lieu of exact solutions. Nemhauser and colleagues (1978) showed that when u is constrained to nonnegative values, the weighted maximum coverage problem is submodular and a $(1 - 1/e)$-approximation to the optimal solution can be computed using a greedy approximation algorithm. Their greedy algorithm, shown previously in Algorithm 1, iteratively constructs $R' \subseteq R$ by adding $r \in R \setminus R'$ that maximally increases the objective given the previously selected elements in R'. In many natural settings the response space R is either exponential or infinite and solving even a single step of such a greedy algorithm may be NP-complete. Kleinberg and colleagues (2004) introduce several approximation algorithms for segmentation problems. In addition to some greedy approximations, which greedily add the response that maximizes the marginal gain like Nemhauser and colleagues' algorithm, they also introduce efficient approximation algorithms that avoid enumerating the response space by exploiting domain-specific assumptions about the utility function's structure.

In many problems, including our domain of extensive-form games, the utility function does not conform to the assumptions required by Kleinberg and colleagues' algorithms. In this chapter, we develop an efficient approximation algorithm by assuming that the utility function's structure admits a specific type of efficient oracle. In the next section, we describe the details of our oracle and demonstrate that extensive-form games admit such oracles.

5.2 Exploiting Structured Utility

In settings where the response space is too large to practically enumerate, algorithms that avoid enumerating the candidate responses are necessary. We examine problems where structure in the utility function u allows the best response for a given set of entities to be efficiently computed despite a prohibitively large response space. Next, we formalize this response oracle, provide examples of problems where such an oracle exists, and show how one can incorporate a response oracle into an efficient greedy agglomerative clustering algorithm.

5.2.1 Response Oracles

In some settings the utility function u can be factored to enable the efficient computation of the response (*i.e.*, column of U) that maximally covers a given set of entities. Formally, given $C \subseteq E$ a **response oracle** $f : 2^E \to R$ is defined as

$$f(C) \in \operatorname*{argmax}_{r \in R} u(C, r).$$

By definition, this means $u(C, f(C))$, the utility obtained by the response oracle on C, is $u(C)$.

We provide some domains where such a response oracle can be computed efficiently, taking time logarithmic in the size of the response space, and then present a greedy algorithm that capitalizes on this oracle.

Example: Extensive-form Games.

Let the entities E consist of observed opponent strategies and let the response space R be the possible strategies that our agent could employ. Even if our agent only considers pure strategies (where a player acts deterministically at each of the game's information sets) the size of the response space is exponential in the number of the game's information sets. The response oracle $f(C)$ in this setting is a best response to the average of the sequence-form representations of the strategies in C. Due to an extensive-form game's tree structure, this can be computed in time linear in the size of the game tree, which is typically polynomial in the number of information sets.

Example: Budgeted Social Choice.

In the case of maximizing social welfare, Lu and Boutilier's (2011) limited choice model of budgeted social choice is equivalent to the segmentation problem optimization in Equation (5.2) (with entities and responses instead called agents and alternatives, respectively). In their formulation, they assume that the responses can be enumerated and use Nemhauser and colleagues' greedy algorithm. Suppose the responses consist of products represented by feature vectors $r = (r_1, \ldots, r_n) \in R_1 \times \ldots \times R_n$ and that the utility function is *separable* such that it can be factored into $u(e, r) = \sum_{i=1}^{n} u_i(e, r_i)$ where $u_i : E \times R_i \to \mathbb{R}$. Then the response oracle $f(C)$ can compute each r_i of the optimal response as $\operatorname{argmax}_{r_i \in R_i} \sum_{e \in C} u_i(e, r_i)$. Instead of enumerating the $\prod_{i=1}^{n} |R_i|$ possible responses in R, the response oracle can be computed efficiently in time $O(|C| \sum_{i=1}^{n} |R_i|)$.

5.2.2 A Greedy Heuristic

Since both exact solutions and algorithms similar to Nemhauser and colleagues' greedy algorithm are infeasible on segmentation problems with a large response space, we propose an alternative greedy approximation algorithm. Like others, our approach acts greedily based on the marginal change in utility, but it does so when considering how to merge clusters in an agglomerative hierarchical clustering algorithm (Ward 1963). Our algorithm is efficient, requiring a polynomial number of oracle queries, provided an efficient response oracle.

In an agglomerative clustering algorithm, one starts with a partition of singletons and builds a solution in a "bottom up" manner by iteratively coarsening the partition. Unlike Nemhauser and colleagues' greedy algorithm, which acted greedily according to the *maximum marginal gain*, we greedily merge clusters that incur the *minimal marginal loss*. This is because the objective value of Equation (5.1) is vacuously maximized by the initial partition of singletons (when the size of the partition is unconstrained). One can view this as a greedy heuristic for the following optimization.

$$\min_{P \in \mathrm{Part}_k(E)} \left[\sum_{e \in E} \max_{r^* \in R} u(e, r^*) - u(P) \right] \tag{5.3}$$

Conceptually, this objective function represents the utility lost by responding to the entities in clusters rather than individually. Note that a partition that optimizes our original objective in Equation (5.1) also optimizes Equation (5.3), and vice versa.

Our algorithm starts with the trivial partition $P_0 = \{\{e\} : e \in E\}$ where every entity is in its own set. On each iteration we coarsen the partition by greedily merging together the two sets in the current partition that incur the minimal marginal loss in the objective value of Equation (5.3). This is repeated until we have a partition that satisfies the cardinality constraint k.

More formally, let Coarse(P) be all the 1-step coarsenings of partition $P = \{C_1, \ldots, C_k\}$:

$$\mathrm{Coarse}(P) = \{\mathrm{Merge}(P, C_i, C_j) : 1 \le i < j \le k\},$$

where

$$\mathrm{Merge}(P, C_i, C_j) = P \setminus \{C_i, C_j\} \cup \{C_i \cup C_j\}$$

is the partition that results from P by merging C_i and C_j. Then, the greedy algorithm can be written as in Algorithm 2. Note that the combination of sets $C_i, C_j \in P$ that produce the optimal coarsening are exactly those for which the marginal loss $u(C_i) + u(C_j) - u(C_i \cup C_j)$ is minimal.

Algorithm 2 Greedy response oracle clustering

```
Require: Set $E$ of entities, a response oracle $f$, utility function $u$, $k$
  Initialization: $G = P_0 = \{\{e\} : e \in E\}$
  while $|G| > k$ do
    $G \leftarrow \text{argmax}_{P \in \text{Coarse}(G)} u(P)$
  end while
  return $G$
```

A naive implementation of this algorithm must compute the marginal loss for all combinations $C_i, C_j \in P$ on each of the $|E| - k$ iterations. This implementation would require $O(|E|^3)$ calls to the oracle. By noticing some structure in the computation, we can use memoization to improve this. After computing the marginal loss for all combinations $C_i, C_j \in P$ of candidate merges on the first iteration, the marginal losses for all candidates can be updated after each merge with at most $O(|E|)$ calls to the response oracle. If C_i and C_j are merged, we need only compute the marginal losses for all pairs of clusters that involve the new cluster $C_i \cup C_j$. This memoization implementation only needs a total of $O(|E|^2)$ calls to the oracle. Additionally, since we know that the marginal loss is always nonnegative, we can lazily update the marginal losses (only evaluating them while no zero loss candidate exists) to further reduce computation. Finally, each marginal loss computation is independent from the others and therefore amenable to parallelization.

Unlike clustering algorithms where the desired number of clusters k needs to be specified in advance (*e.g.*, Lloyd's algorithm for k-means), our greedy algorithm's iterative and deterministic nature means that it could be run a single time for $|E|$ iterations, reporting the partitions and objective value on each iteration. This enables users to directly evaluate the trade-off between the objective and the number of clusters.

In the remainder of this chapter, we evaluate our greedy algorithm both theoretically, proving approximation bounds on the solution quality, and empirically by clustering agent strategies in toy poker games. We begin with our theoretical analysis.

5.3 Theoretical Results

Our theoretical analysis examines the worst-case behaviour of our greedy clustering algorithm. Theorem 1 establishes a lower bound on the utility of a clustering produced by our greedy algorithm, relative to the optimal clustering into k sets. Theorem 2 then shows that this lower bound is tight (within a factor of 2) and cannot be improved substantially. Although we provide proof sketches of these results, we refer readers to the full proofs in "Decision-theoretic Clustering of Strategies" as they were provided by coauthors Deon Nicholas and Csaba Szepesvári.

In this section we fix E and we denote its cardinality by m. We begin with the following result bounding the worst-case performance of our greedy algorithm:

Theorem 1. *Let u be a nonnegative valued utility function, $1 \le k \le m$. Then*

$$u(G_k) \ge \max\left(\frac{1}{k}, \frac{k}{m}\right) u_k^* \ge \frac{1}{\sqrt{m}} u_k^*,$$

where G_k is a k-element partition of E returned by the greedy algorithm.

Proof sketch. Note that the second inequality follows trivially from

$$\max\left(\frac{1}{k}, \frac{k}{m}\right) \geq \min_{s>0} \max\left(\frac{1}{s}, \frac{s}{m}\right) = \frac{1}{\sqrt{m}}.$$

Hence, it remains to prove the first inequality. We prove this by separately showing that $u(G_k) \geq \frac{k}{m} u_k^*$ and that $u(G) \geq \frac{1}{k} u_k^*$.

To establish $u(G_k) \geq \frac{k}{m} u_k^*$, we first prove a lower bound on the utility of a single coarsening step involving $C = \operatorname{argmin}_{C \in G} u(C)$, the lowest utility element of the partition G. Since the greedy algorithm uses the best possible coarsening on each step, this also provides a lower bound on the utility for any single coarsening step of the greedy algorithm. Finally, we apply this result $(m-k)$ times to provide a lower bound on the utility of G_k.

Next we establish the bound $u(G) \geq \frac{1}{k} u_k^*$ on the utility for any partition $G \in \operatorname{Part}(E)$. Observe that for $C \subseteq E$, $u(G) \geq u(C)$ since u is nonnegative and the best response for C could be used for each $C' \in G$. Furthermore, for any optimal k-element partition, P_k^*, the element $C \in P_k^*$ with highest utility must contribute at least $\frac{1}{k} u(P_k^*)$. □

The next result establishes that the bound in Theorem 1 cannot be substantially improved:

Theorem 2. *Let k be a positive integer. For each $\varepsilon > 0$, there exists a nonnegative utility function on a set of entities E of $m = k^2$ elements and on a set R of $k^2 + k$ responses such that if G_k is the k-element partition returned by the greedy algorithm when fed with k and u then $u(G_k) - \varepsilon \leq \frac{2}{\sqrt{m}} u_k^*$.*

Proof sketch. We will construct a matrix of size $k^2 \times (k^2 + k)$ holding the values of the utility function. Let $E = \{1, \ldots, k^2\}$, I denote the $k \times k$ identity matrix, $\varepsilon > 0$ and define the $k \times k$ matrix Q_ε by

$$Q_\varepsilon = \begin{pmatrix} 2+\varepsilon & \varepsilon & \ldots & \varepsilon \\ \varepsilon & 2+\varepsilon & \ldots & \varepsilon \\ \vdots & \vdots & \ddots & \vdots \\ \varepsilon & \varepsilon & \ldots & 2+\varepsilon \end{pmatrix}.$$

Finally, as in Section 5.1, the $k^2 \times (k^2 + k)$ matrix representing the utility function u is given by

$$U = \begin{pmatrix} I & Q_\varepsilon & 0 & \ldots & 0 \\ I & 0 & Q_\varepsilon & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I & 0 & 0 & \ldots & Q_\varepsilon \end{pmatrix}.$$

Let $C_1 = \{1, \ldots, k\}$, $C_2 = \{k+1, \ldots, 2k\}$, …, $C_k = \{k^2 - k + 1, \ldots, k^2\}$. Note that $\{C_1, \ldots, C_k\} \in \operatorname{Part}_k(E)$. We claim that the greedy algorithm returns the k-element partition $G_k = \{C_1, \ldots, C_k\}$ while the optimal k-element partition is $P_k^* = \{\{1, k+1, 2k+1, \ldots, k^2-k+1\}, \{2, k+2, 2k+2, \ldots, k^2-k+2\}, \ldots, \{2k, 3k, \ldots, k^2\}\}$ when ε is sufficiently small. □

While these theoretical results provide guarantees about our greedy algorithm's *worst-case* performance, it can perform much better in practice. In the next section, we empirically evaluate our greedy algorithm's practical performance by clustering agent strategies in two toy poker games.

5.4 Strategy Clustering

We demonstrate the performance of our greedy heuristic algorithm by clustering agent strategies for extensive-form games. This experiment not only provides a valuable evaluation of our algorithm, but it also highlights the potential value of such decision-theoretic clustering techniques in generating a compact set of responses for a portfolio. We begin our empirical analysis by describing the design of our empirical evaluation, including how we cast this problem as a segmentation problem.

5.4.1 Experimental Design

To evaluate our greedy decision-theoretic clustering algorithm, we contrast its performance with a k-means clustering algorithm in the two toy poker games discussed in Section 2.2: Kuhn poker and Leduc hold'em. We start by describing how clustering agents in an extensive-form game can be cast as a segmentation problem, and then detail our benchmark k-means algorithm before moving on to our empirical data.

In this setting, we seek a partition of a set E of static agent strategies that optimizes Equation (5.1). In our experiments, we generate the agents in E by sampling 200 static strategies uniformly at random from the strategy space. Though not constructed explicitly, the utility matrix U can be viewed as having a column for each possible strategy and entries $u(e, r)$ corresponding to the expected utility of playing strategy r against static agent e. As mentioned previously, the response oracle $f(C)$ for a set of strategies C is the best response to the average of the sequence-form representations of the strategies in C.

Each of our experiments contrasts our greedy algorithm with the standard k-means clustering algorithm, *i.e.* Lloyd's algorithm (Lloyd 1982), using the sequence-form representation (Koller, Megiddo, and von Stengel 1994) of an agent's strategy as its feature vector. We initialize the cluster centroids using the k-means++ algorithm (Arthur and Vassilvitskii 2007). Note that while the locally optimal clustering found by Lloyd's algorithm may be arbitrarily bad in terms of the k-means objective, initializing with k-means++ provides an approximation guarantee on the solution quality. Despite this, the stochasticity of the k-means initialization impacts which local optimum is found. In our experiments, k-means is restarted 50 times and the clustering with the best k-means objective (*i.e.*, minimal within-cluster sum of squared Euclidean distances) is reported.

Note that we use the sequence-form representation as opposed to a behavioural strategy representation because behavioural strategies allow for some unintuitive behaviour. To illustrate, consider a strategy with an information set where one action is never chosen. In sequence-form, the realization weights for the entire subtree under that action are zero, but a behavioural strategy may have arbitrary distributions there. This could allow two strategies that are identical in sequence-form to be arbitrarily spatially distant in a behavioural strategy representation. Similarly, two behavioural strategies that differ only at a single information set would be very spatially similar while having very distant sequence-form representations. Although Euclidean distance between behavioural strategies has been used to measure similarity between strategies previously in poker games (Ganzfried and Sandholm 2011), we opted for sequence-form representations because of this unintuitive behaviour. That said, note that in undominated Kuhn poker, all action sequences use at most one of the aforementioned parameters. Thus, in this particular case, the sequence-form representation of a strategy happens to coincide with the behavioural strategy representation.

5.4.2 Empirical Results

We begin the analysis of our greedy clustering algorithm by examining both its qualitative and quantitative performance when clustering player two's strategies in undominated Kuhn poker.

Kuhn Poker

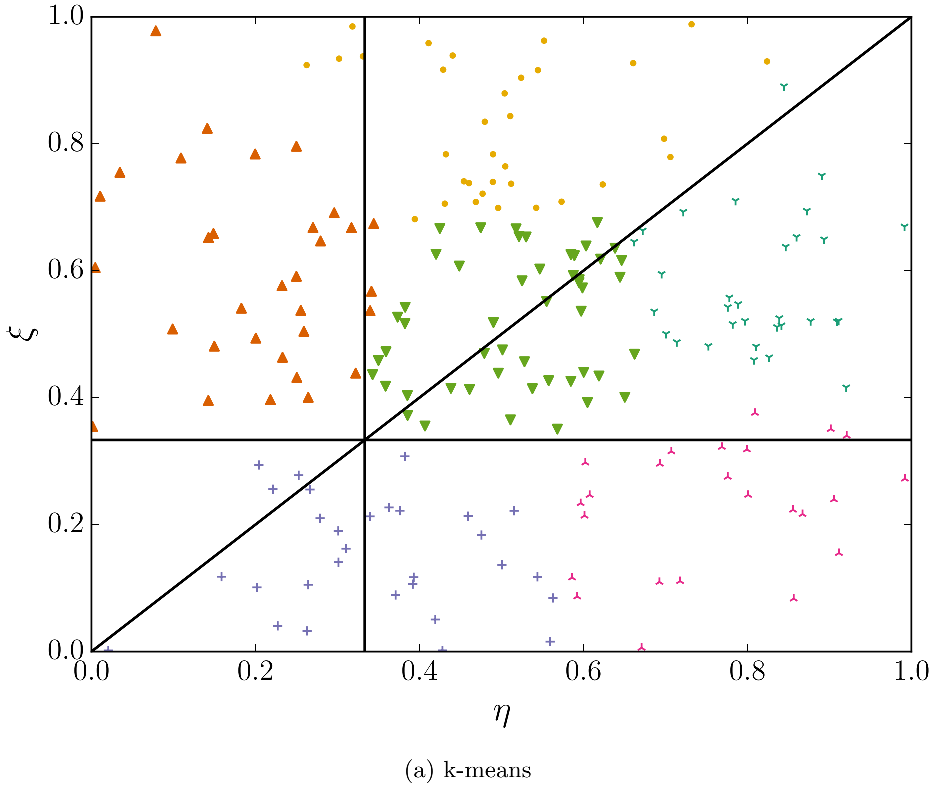
Figure 5.3 contrasts clusterings produced by k-means and our greedy algorithm to demonstrate the qualitative differences in these clustering approaches. These figures visualize each of the 200 agent strategies according to their two parameters, η and ξ. The marker shape and colour correspond to which cluster each strategy is assigned to. As with the rock-paper-scissors example, the boundary lines between each of the best response regions are plotted. This partitions the strategy space into six regions each with a distinct best response. The k-means and greedy algorithms were run with $k = 6$ clusters as this is sufficient to optimally partition the strategies. Figure 5.3a illustrates how k-means tends to produce relatively spherical clusters of similar size. Unsurprisingly, k-means' optimization of spatial distances results in clusters that fail to respect these boundaries. In contrast, our greedy algorithm (Figure 5.3b) is able to exactly partition the agents according to the best response regions.

Next, we examine the quantitative performance of these algorithms in terms of the utility lost due to responding to agents in clusters rather than as individuals (as in Equation (5.3)). Figure 5.4a shows the loss incurred by k-means and our greedy algorithm as we vary the number of clusters for undominated Kuhn poker. In this domain we also compute the optimal clustering through enumerating all possible combinations of k of the six best responses and then inducing the corresponding partition as per Lemma 2. Results have been averaged over 50 trials each sampling a new set of 200 agents. The trend lines show the mean value for the loss and the surrounding shaded region indicates the 95% confidence interval (which is occasionally difficult to see as it is smaller than the line width). Values are in milli big blinds per game (*i.e.*, thousandths of the initial ante). While the greedy algorithm manages to achieve zero loss once allowed the six clusters required to guarantee the points can be properly partitioned, k-means is unable to reach zero loss even after being given twice as many clusters. This result also highlights how our greedy algorithm can obtain considerably more of the optimal clustering's utility than is guaranteed by our worst-case approximation bound.

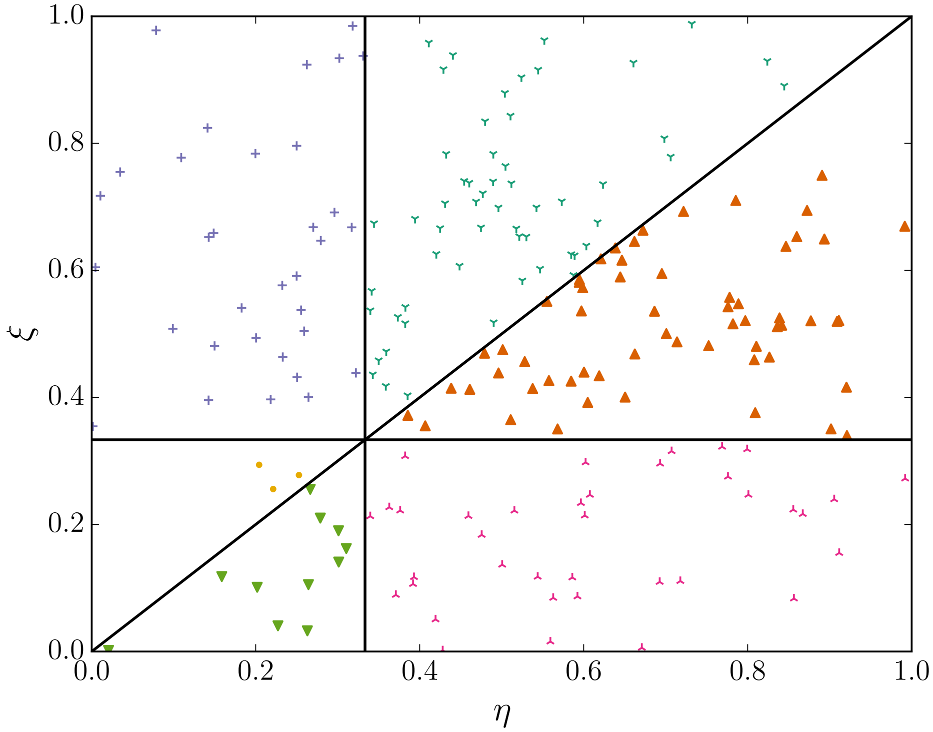
Leduc Hold'em

Though Kuhn poker provides a convenient domain for visualizing strategies and best response regions, clustering agents in such small domains can be done through direct analysis of the game or brute force enumeration of a player's pure strategies. Leduc hold'em better demonstrates the value of a response oracle as the game is sufficiently large that such enumeration is infeasible. Figure 5.4b shows similar quantitative performance results for the domain of Leduc hold'em where strategies for player one are being clustered. Note that the results in Figure 5.4 exploit the fact that the greedy algorithm's performance can be computed for each k with little additional computation (though Figure 5.4b omits values over 64). It is clear in these results that our greedy clustering algorithm substantially outperforms k-means. In particular, the greedy algorithm achieves approximately the same performance with 7 clusters as k-means does with 64.

Finally, it is interesting to note the rate of improvement as we allow for more clusters. Observe that while the greedy algorithm initially improves rapidly as we increase the number of clusters, the rate of improvement quickly levels off and leaves a very long tail compared to Kuhn poker. This is likely due both to the increased complexity of the game and also the uniform random sampling of the strategy space. Unlike Kuhn poker where we would expect each best response region to contain multiple of the 200 sampled strategies, the more complex strategy space of Leduc hold'em likely means any given best response region is (at best) sparsely sampled. If the agents being clustered were covered by relatively few of a game's best response regions, then this long tail may not be present.

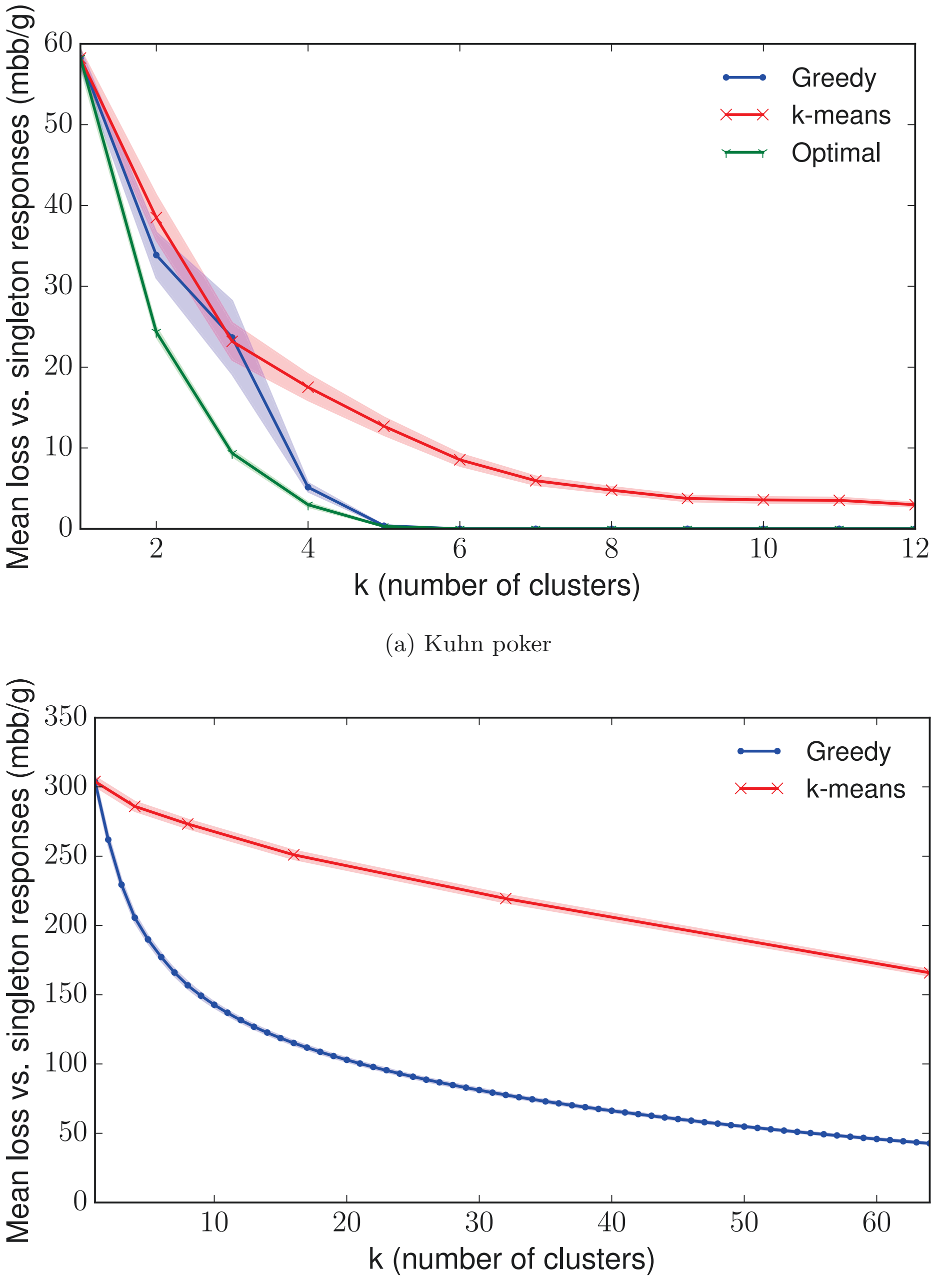


(a) k-means



(b) Greedy agglomerative clustering

Figure 5.3: Clusters of player two Kuhn poker agents found using k-means clustering over the sequence-form representations and our greedy heuristic algorithm ($k = 6$). Marker shape and colour indicate cluster membership.



(a) Kuhn poker

(b) Leduc hold'em

Figure 5.4: Performance of different clustering techniques in toy poker domains.

5.5 Summary

Agents seeking to maximize their utility may be able to improve their performance by exploiting models of other agents or entities in their environment. In these settings, clustering techniques can be beneficial for extracting similar groups of entities. Despite the ubiquity of spatial clustering techniques, spatial similarity may be insufficient for capturing similarity in how an agent should respond to these groups. Instead, practitioners with utilitarian clustering objectives, such as optimizing an implicit modelling agent's portfolio, may prefer to explicitly optimize the utility of an agent's responses to the clusters. Although work related to segmentation problems provide techniques to optimize for such actionable clusters, these techniques may be computationally infeasible for domains with large response spaces.

We introduced an efficient greedy algorithm for this type of decision-theoretic clustering that can exploit the structure of certain domains. We sketched proofs for worst-case approximation bounds on the quality of solutions produced by our greedy algorithm. Finally, we showed how to apply this technique to extensive-form games, and empirically demonstrated the value of this approach by comparing it to k-means for clustering agent behaviours in two toy games of poker. In the next section, we present our final contribution: an end-to-end application of the implicit modelling approach for producing adaptive agents in human-scale extensive-form games.

Chapter 6

Human-scale Implicit Modelling

In Chapter 3, we presented a novel characterization of agent modelling techniques that contrasted the explicit and implicit modelling approaches, highlighting the challenges of each. We argued that it was unclear how agent modelling practitioners could adequately address many of the challenges inherent in using explicit models online for human-scale domains. Furthermore, we provided some general ideas for how agent modelling practitioners could build and use a portfolio to produce an implicit modelling agent for such domains. However, we deferred specific details and empirical validation of our implicit modelling agent so we could use concepts from our earlier contributions in building our portfolio.

In this chapter, we detail how we address the challenges of using an implicit model in human-scale extensive-form games and empirically validate the approach. Poker provides a natural domain for experimental validation of these techniques. Since human experts are known for quickly adapting to other players, this sets a high bar for agent modelling techniques to strive for. We empirically validate our approach using variants of Texas hold'em poker, a complex domain where explicit modelling has generally been unsuccessful against sophisticated agents.

With a focus on statistical significance, yearly competitions since 2006, and numerous competitors worldwide, the Annual Computer Poker Competition (ACPC) is the premier venue for computer poker research. In addition to running the yearly competition, the ACPC provides public access to the results, logs, and participant descriptions from past events on its website (ACPC 2015). Furthermore, following each competition, the ACPC hosts a benchmark server that participants can use to evaluate against agents from that year's competition. The ACPC organizers have our gratitude for continuing to provide these services to the community, as all of them were central in the construction and evaluation of our implicit modelling agents. We present results against agents from several prior years of ACPC events in each of the three Texas hold'em poker variants used in the competition (described in Section 2.2). Specifically, we evaluate implicit modelling agents in the competition's total bankroll events, which highlight agents' abilities to model and adapt.

We begin our empirical validation by using the domain of heads-up limit Texas hold'em for a case study of implicit modelling in human-scale settings. This case study presents both controlled experiments, where we evaluate implicit modelling agents relative to several baseline approaches, and experiments against agents from the 2011 ACPC benchmark server. Prior to our first set of results, we detail our implicit modelling agent end-to-end: from building response strategies, to selecting responses for the portfolio, and finally using online learning algorithms to combine the portfolio's strategies during online interaction. Subsequent evaluation provides more anecdotal validation by examining the performance of agents that were actually submitted to prior competitions using the ACPC's published results. We present results from each variant, separated by the competition year, beginning with heads-up limit Texas hold'em, followed by three player limit, and finally heads-up no-limit. Though details of our implicit modelling agent's portfolio varies between the domains and competition years, most high-level algorithm choices remain the same as our case study agent. Prior to each result, we provide details specific to that particular agent's portfolio.

6.1 Case Study: Heads-up Limit Texas Hold'em

Our first empirical results provide a detailed analysis of our implicit modelling framework in the domain of heads-up limit Texas hold'em poker. At the inception of the Annual Computer Poker Competition in 2006, heads-up limit Texas hold'em was the sole event. Growing from its start of five competitors in 2006, this long-standing ACPC event drew numerous competitors from around the world. This made the competition an ideal setting for our agent modelling task, as it provided a long history of competitions that could furnish us with data for diverse and increasingly sophisticated agents.

We show that our implicit modelling approach outperforms several other baseline approaches and that an agent using the framework would have won the 2011 Annual Computer Poker Competition's total bankroll competition. The winner of this event is the agent with the highest expected winnings against all other competitors in the event[1]. The implicit modelling agent design and empirical results presented here were originally published as "Online Implicit Agent Modelling" (Bard et al. 2013).

6.1.1 Agent Design

As discussed in Chapter 3, there are several key questions that must be addressed in constructing an implicit modelling agent. Specifically, how should practitioners build an effective portfolio and subsequently use the portfolio online? We discuss each part of this question in turn, providing the techniques used in this case study and for parts of our subsequent ACPC submissions.

Building a Portfolio

With performance in the Annual Computer Poker Competition as our goal for these experiments, we chose to use data from past competitors to generate our responses. To keep training data and testing opponents separated, we used logs from the 2010 ACPC for generating our response strategies. This provided data for 13 different agents, each with a total of at least 8.4 million full information hands (*i.e.*, no missing card information) against other competitors from that year.

Robust Responses. The data for each individual agent was used with Johanson and Bowling's (2009) DBR technique to compute a robust response. As we are dealing with a human-scale game, DBR requires an abstraction choice for both the opponent and the response strategy. As we showed in Section 4.2, choosing these abstractions asymmetrically can yield substantially better DBR strategies. To mitigate sparsity in the opponents' frequentist models, we use the coarse 5-bucket perfect recall percentile $E[HS^2]$ card abstraction introduced in Section 4.1.1. Abstractions for both the robust response and the opponent's response used a card abstraction somewhat smaller than the strongest entries in the actual 2011 competition. The abstraction clusters cards into 9,000 buckets using k-means for each round after the preflop, forgetting earlier buckets. On the flop and turn, clusters are formed based on the earth mover's distance between hand strength distributions, while opponent cluster hand strength features are used for clustering on the river. Johanson and colleagues' (2013) provide further details and evaluation of this imperfect

[1] Following the 2012 ACPC, the evaluation of the total bankroll event changed. We defer discussion about that change and its ramifications until Section 6.2.1

recall abstraction under the name “IR KE-KO 169-9000-9000-9000”. DBR also requires the specification of the $P_{\max}$ parameter to control the trade-off between the response's exploitation and exploitability. We tuned this parameter to generate responses that were exploitable for approximately 100 mbb/g (milli big blinds per game) by a best responding agent trained in the “IR KE-KO 169-9000-9000-9000” abstract game. This threshold was kept relatively low to ensure that any of the responses could be used without substantial risk.

Pruning Responses. While we can generate a robust response from every past interaction, it may not be wise to include all such responses in the portfolio. An overly large portfolio has two drawbacks. First, it adds computational burden in order to estimate the utilities of every strategy in the portfolio after every hand. Second, and more importantly, both theoretical bounds and empirical practice of bandit-style algorithms (such as Exp3G) show regret growing with the number of available bandit arms. Too many arms simply requires too much exploration before exploitation can reliably occur. Furthermore, each additional response may not be adding much to the overall exploitive power of the portfolio if other similar responses are already included. Finding a manageably-sized portfolio that still achieves broad exploitation possibilities is the final step in our portfolio construction.

Generating a portfolio that efficiently covers the space of other agents' behaviours can be posed as the same weighted maximum coverage optimization that we discussed in Chapter 5. Specifically, we would like to find some small subset R' of our entire set of generated responses R that retains as much of the exploitive power as possible. However, unlike the previous segmentation problems, we lack a “ground truth” objective. This is not only due to uncertainty about which agents we will eventually encounter, but also because we cannot compute a given portfolio's exploitive power without using our dynamic agent to interact with the other agents. As such, we will need a proxy objective for the portfolio's exploitive power.

We define this proxy objective function on portfolios to be the total expected utility the portfolio would obtain if the implicit modelling agent acted according to the best response in the portfolio when playing each of a given field of agents (*i.e.*, the response with the maximum expected utility for the agent at hand). Yet, we still need a field of agents to formally define our objective. This is where we can use the mimic strategies that are generated as a by-product of DBR. Our objective is then the total expected utility achieved against all of the generated mimic strategies if we can optimally choose the portfolio's utility-maximizing response for each mimic.

This is similar to the maximum coverage view of our earlier strategy clustering problem, shown in Equation (5.2), except the space of responses is restricted to the DBR strategies. As in that problem, when the number of responses is sufficiently small we can optimally solve the optimization through brute force enumeration. When such an optimal solution is infeasible, Nemhauser and colleagues' (1978) greedy heuristic in Algorithm 1 can also be used to provide good approximate solutions.

Alternatively, one could attempt to use the decision-theoretic clustering technique we introduced in Chapter 5 to reduce the size of the portfolio. Similar to our proxy objective, we could cluster the agents' mimics and subsequently build DBRs to each cluster. This would allow us to cluster the strategies according to the full space of responses rather than merely the DBR strategies. However, as this approach requires a quadratic number of best response computations, it rapidly becomes infeasible for human-scale problems. Unfortunately, since we did not realize the connection between the partitioning and maximum coverage views of segmentation problems until after the 2011 benchmark server was retired, our results do not employ our decision-theoretic clustering technique.

If computational resources limit the number of responses then this can serve as our cardinality constraint and stopping condition. Alternatively, the marginal increase in value for including each additional response can provide a guide: stopping once the diminishing returns becomes too small. Our experimental results show the performance of our implicit modelling approach using two different portfolios: a portfolio of all of the responses (except the responses to our own 2010 submission to the ACPC), and a smaller four-response portfolio generated by this greedy approximation to the maximum coverage problem. With portfolio construction completed, we move on to addressing how our implicit modelling agents use their portfolio during online interaction.

Using a Portfolio

We apply Exp3G (Kocsis and Szepesvári 2005) directly to our task of combining the responses in our portfolio based on each response's expected total reward[2]. In our experiments, we make two small changes to Exp3G. First, rather than acting according to a single expert sampled from Exp3G's distribution, we act according to the weighted mixture of the experts. Note that since our expert responses are extensive-form strategies instead of a distribution over single actions, we must average the strategies' action sequence probabilities when mixing the experts. Second, instead of uniform exploration over the experts, we force the weight of each expert to be at least some minimum value, bounding each expert's weight in the mixture away from zero. Both approaches guarantee the acting strategy has non-zero probability on every action sequence played by any response in the portfolio.

We chose Exp3G parameter settings by performing experiments against our generated mimic strategies using the small portfolio. One of the best settings used a temperature parameter of $\eta = 0.025$ and a minimum probability of 2%, although the performance was largely insensitive to these parameter choices, with broad ranges of parameters giving similar results. After a smaller experiment verifying this choice of parameters using the large portfolio, we used these parameter settings in all subsequent experiments.

Since Exp3G leaves the utility estimation mechanism unspecified, we must still choose an estimation procedure. Our agents' estimators use Bowling and colleagues' (2008) combination of off-policy importance sampling and imaginary observations. Their work explored the effects of various types of imaginary observations and value functions for strategy evaluation. For our experiments we create observations for all possible private cards and early folding opportunities. Early folds are provably unbiased, and although the all-cards technique can create bias under partial information (due to card-replacement effects and not knowing which cards the other agent holds) prior results in heads-up limit Texas hold'em suggest this bias is small while providing substantial variance reduction (*viz.*, Bowling et al. 2008, Table 3). Furthermore, we use the basic value function (*i.e.*, the "money" exchanged at the end of a game), as opposed to the DIVAT or bet-call DIVAT techniques examined by Bowling and colleagues (2008), to avoid introducing additional bias in our partial information setting[3]. Finally, note that our choice of exploration parameters ensures Exp3G will sample according to a strategy that mixes between all of the strategies in the portfolio, thereby avoiding potential bias from off-policy importance sampling. We depict the entire process of our implicit modelling agent in Figure 6.1.

6.1.2 Empirical Results

While the ultimate validation is comparing the approach in the context of the 2011 ACPC entrants, we begin our experimental validation in a number of simpler settings. The first is a

[2] Throughout our original exposition in "Online Implicit Agent Modelling" (Bard et al. 2013), we related our approach to a slightly modified version of Exp4. Though still accurate, Exp3G is more closely related to our approach and so we use it here.

[3] Note that the DIVAT-based estimators are only unbiased when full information is available, such as evaluating a match post hoc using complete logs. When evaluating a portfolio *online*, full information would not be available. Applying additional variance reduction techniques that would be unbiased for an online setting is a direction for future work, which we discuss in Section 7.2.2

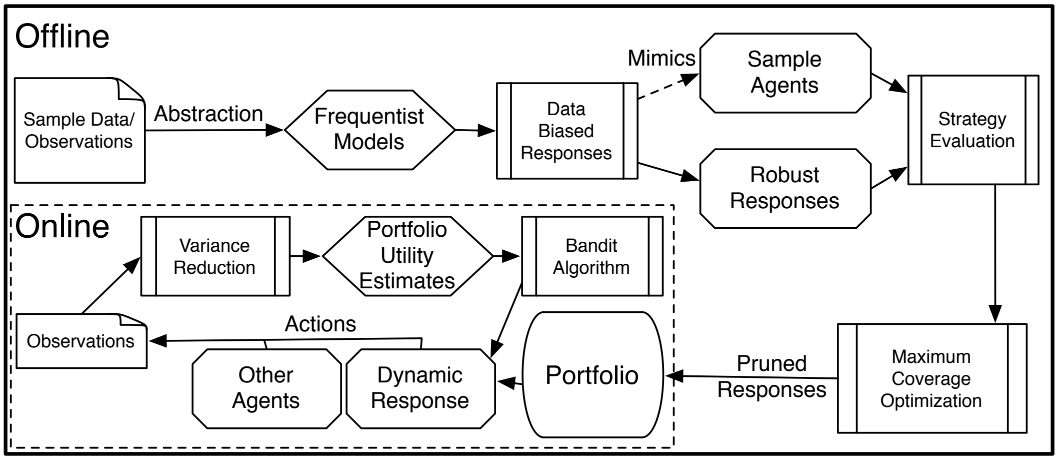


Figure 6.1: Implicit modelling process. The portfolio is constructed offline from robust responses to agent models and pruned with a (submodular) maximum coverage optimization. It is then used online by a bandit algorithm in tandem with variance reduction techniques.

comparison against the four mimics for which the small portfolio's responses were designed. This experiment gives us an idealized case where we avoid any concerns about being unable to respond to one of the opponents. Furthermore, because the opponents include two of the most exploitable agents from the 2010 ACPC, this experiment is an ideal scenario for exploitation. In all of the following stacked bar charts, the total bar height indicates the expected winnings against the field of opponents while the bar's composition specifies the proportion of the mean won by playing against each specified mimic. Note that the vertical order of the bar's components is the same as the order in the legend. Expected winnings are in milli big blinds per game (mbb/g). The match length for all experiments was 3,000 hands: the same length as the 2011 ACPC event. 95% confidence intervals on all of the expected winnings values are ±7mbb/g or smaller, except where noted. Essentially, visible differences in the graphs are statistically significant.

Versus Small-Portfolio Mimics

The results for this simple setting are in Figure 6.2. Our implicit modelling agent using the small portfolio (labelled Small-Portfolio) outperforms our range of other baselines. First, note that its winning rate is nearly double that of a fixed Nash equilibrium strategy using a considerably larger abstraction. Furthermore, the Small-Portfolio agent improves upon any individual response from the portfolio (ASVP, GS6_iro, LittleRock, longhorn) by at least 19.9%, demonstrating that it is able to tailor its strategy online to the opponent at hand. It also outperforms the Small-Static agent: a baseline also built using a DBR, but responding to a single aggregate model built with the same data used for the small portfolio's models. This suggests that the mimics are exploitable in at least partially independent ways and that modelling them as a single aggregate player harms the response's exploitive power.

Small-Portfolio also improves on an implicit modelling agent that uses upper confidence bounds (UCB) instead of our modified Exp3G algorithm to select from the small portfolio. This is despite the fact that stochastic bandit algorithms like UCB are, in one sense, better tailored to this particular experiment as they correctly model the reward distributions produced by the static mimics. However, since UCB selects and acts according to a single strategy from the portfolio rather than a mixture, we cannot ensure that the support of the strategy being played is a superset of the supports for the portfolio's individual responses. Therefore we cannot estimate the utilities of the off-policy strategies in the portfolio without

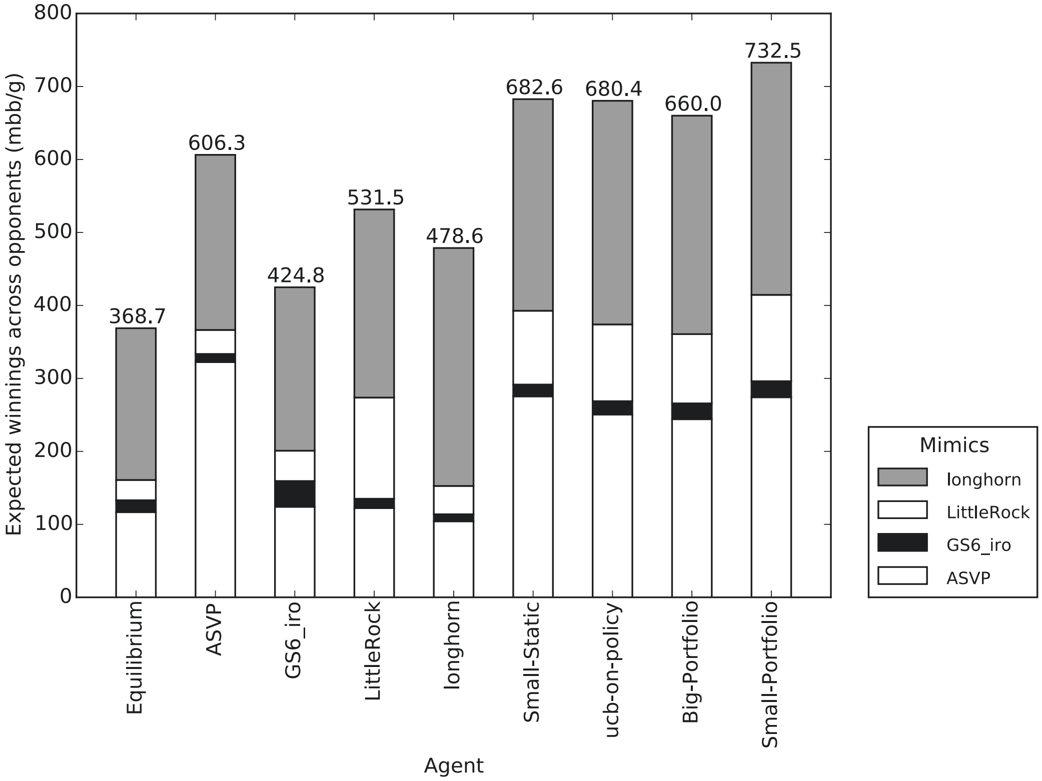


Figure 6.2: Performance versus mimics corresponding to the four responses in the small portfolio. Bar components are in the same order as the legend.

potentially introducing bias. Note that this UCB based agent exhibited higher variance in its results, and a 95% confidence interval on its results are approximately ±13 mbb/g.

Finally, the smaller portfolio also outperformed the larger portfolio (Big-Portfolio). This demonstrates that even though Big-Portfolio contains a superset of the responses, there is a potential learning cost associated with including extraneous strategies in the portfolio.

Versus Big-Portfolio Mimics

Next, we evaluate the agents against the entire field of 2010 ACPC mimics. In contrast to the last experiment, the Small-Portfolio agent no longer has a response associated with every opponent, while the Big-Portfolio agent still does. The results are in Figure 6.3. Once again, the small portfolio agent outperforms the equilibrium, now by approximately 65%. Against this field, both Exp3G-based implicit modelling agents outperform the aggregated Small-Static baseline agent and the UCB based implicit modelling agent. Despite the reduced number of responses, the Small-Portfolio agent's performance is within noise of the Big-Portfolio agent. These empirical results support our intuition for the benefits of pruning back the portfolio to a manageable size.

Versus 2011 ACPC Agents

Finally, we validated the implicit modelling approach using the 2011 ACPC benchmark server. The benchmark server allows previous competitors to run matches against submissions from that year. Except for one agent that failed to run on the server, all other competitors were available. Table 6.1 shows the performance of the Big-Portfolio and Small-Portfolio

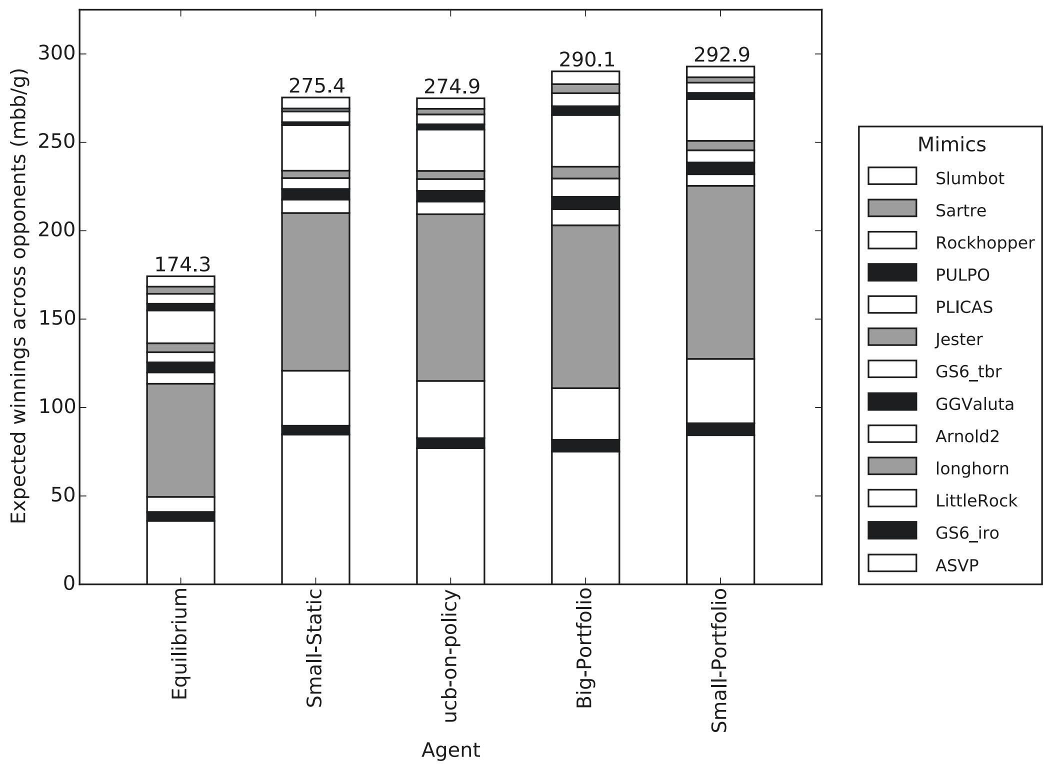


Figure 6.3: Performance versus mimics corresponding to the responses in the large portfolio. Bar components are in the same order as the legend.

	AVG
Calamari	276 ± 4
Sartre	270 ± 6
Hyperborean	224 ± 6
Slumbot	214 ± 6
Feste	213 ± 5
ZBot	205 ± 6
Patience	204 ± 6
2Bot	187 ± 6
LittleRock	183 ± 6
GGValuta	147 ± 6
AAIMontybot	−31 ± 12
RobotBot	−36 ± 9
GBR	−54 ± 13
player.zeta	−189 ± 16
Calvin	−246 ± 12
Tiltnet	−287 ± 10
POMPEIA	−541 ± 6
TellBot	−738 ± 16

(a) versus 2011 ACPC

	AVG
Big-Portfolio	290 ± 11
Calamari	275 ± 5
Sartre	268 ± 8
Feste	211 ± 6
Slumbot	209 ± 7
Patience	207 ± 7
ZBot	205 ± 8
2Bot	188 ± 8
LittleRock	182 ± 9
GGValuta	142 ± 7
AAIMontybot	−28 ± 17
RobotBot	−42 ± 14
GBR	−58 ± 16
player.zeta	−205 ± 21
Calvin	−243 ± 13
Tiltnet	−295 ± 13
POMPEIA	−548 ± 9
TellBot	−757 ± 17

(b) versus Big-Portfolio

	AVG
Small-Portfolio	317 ± 5
Calamari	277 ± 4
Sartre	272 ± 6
Slumbot	216 ± 6
Feste	215 ± 5
ZBot	207 ± 6
Patience	205 ± 6
2Bot	187 ± 6
LittleRock	185 ± 6
GGValuta	147 ± 6
AAIMontybot	−33 ± 12
RobotBot	−41 ± 10
GBR	−61 ± 13
player.zeta	−203 ± 15
Calvin	−252 ± 12
Tiltnet	−300 ± 9
POMPEIA	−553 ± 6
TellBot	−783 ± 15

(c) versus Small-Portfolio

Table 6.1: 2011 ACPC total bankroll results for heads-up limit Texas hold'em. Results include those from original ACPC matches and matches played on the benchmark server against the Big-Portfolio and Small-Portfolio implicit modelling agents. 95% confidence intervals are shown and values are in milli big blinds per hand.

agents from the previous two experiments contrasted with the competition's original results.

Our original submission to the competition in 2011 placed third with a total bankroll of 224 mbb/g with a 95% confidence interval of 6 mbb/g. Note that these results for the original competitors have been recomputed to exclude the agent that failed to run on the benchmark, although it did not change the outcome of the competition. To compute the results for our implicit modelling agents, data from our original agent "Hyperborean" (Hyperborean-2011-2p-limit-tbr) was excluded and new data was substituted for Hyperborean by playing matches on the benchmark server with our implicit modelling agents, Big-Portfolio and Small-Portfolio. The values for each agent's total bankroll in the modified competitions were recomputed using the data from these new matches combined with the remaining original data.

In this experiment both Big-Portfolio and Small-Portfolio do not have any responses tailored for the specific opponents since all responses were trained on data from 2010 ACPC agents. First, observe that both implicit modelling agents would have won the competition. Although the Big-Portfolio agent did not win by a statistically significant margin, Small-Portfolio wins the event by greater than the 95% confidence margin. These results demonstrate that the implicit modelling framework enables modelling and improved utility even against an unknown field of agents. Moreover, while the technique still outperforms other agents with a larger portfolio, pruning the portfolio of redundant or low value responses can improve performance further still. We continue our evaluation by examining the performance of implicit modelling agents that we submitted to subsequent ACPC events.

6.2 Annual Computer Poker Competition Results

The remainder of the chapter examines results from past ACPC total bankroll events. Unlike the previous results, which examined a variety of baselines and implicit modelling approaches, each of these ACPC results examine the performance of a single deployed agent that used techniques from this work. In the ACPC's total bankroll events (occasionally abbreviated as TBR), the winner is the agent with the highest expected winnings against all other competitors in the event. However, the results of the 2012 ACPC motivated changes in the total bankroll winner determination mechanism, which we discuss following our presentation of the 2012 results. In contrast, the ACPC's bankroll instant runoff (or IRO) events rank competitors by iteratively computing total bankroll scores for the field of agents and removing the agent with the worst score from the field. This winner determination mechanism attempts to elicit strong equilibrium agents. In addition to examining the performance of our implicit modelling agents in the total bankroll events, we also compare the performance of our implicit modelling agent with the instant runoff Hyperborean entries submitted by the University of Alberta's Computer Poker Research Group.

Agents submitted to the ACPC must abide several technical restrictions designed to provide as level a playing field as possible. Agents must run on machines provided by the ACPC and are restricted in how much time they can spend computing. Although this does not prevent competitors from exploiting superior resources prior to the competition, all competitors have access to the same quality of hardware and disk space during the competition itself. For each match, agents are allowed no more than 10 minutes during startup, and no more than an average of 7 seconds per hand during play (*e.g.*, 21,000 seconds over a 3,000 hand match). This time can be spent at the agent's discretion over the span of the match provided they take no longer than 10 minutes to make any single action. In addition to making the competition practical to evaluate, these time restrictions also ensure agents – including our implicit modelling agents – act at a pace appropriate for play with humans.

To evaluate the agents, duplicate matches between each combination of agents were played, with each agent's memory being wiped prior to each match. Matches consisted of 3,000 hands, as in our case study, except for the 2013 3-player limit Texas hold'em competition where matches were 1,000 hands. Common random number seeds for the cards were also used to reduce variance. To streamline our exposition, we highlight specific data from past competitions as necessary and reference the appropriate full competition results included in Appendix A. Furthermore, we only provide high-level details of the strategies used in our agents' portfolios, with specific details (*e.g.*, abstraction choices) presented in Appendix B. Our first implicit modelling agent submitted to the ACPC continues where our heads-up limit Texas hold'em case study ended.

6.2.1 Heads-up Limit Texas Hold'em

Our ACPC results for heads-up limit Texas hold'em examine the performance of our implicit modelling agents against competitors spanning from 2012 to 2014, which is the last year the event was held. We discuss the competitions chronologically: presenting our agent and salient results along with some discussion about changes to the competition's format.

2012

Agent Design. The Hyperborean agent submitted to the 2012 total bankroll competition consisted of seven abstract strategies:

1–2. Counter-strategies to opponents that always raise or always call

3. An abstract Nash equilibrium approximation

4–7. Data biased responses using coarse asymmetric models of particular opponents seen in the 2010 (ASVP) or 2011 ACPC (RobotBot, TellBot, Tiltnet)

While it may seem naive to expect opponents to use an always raise or always call strategy, prior competitions had agents that appeared to act this way (potentially due to bugs in the agent's code). The responses to these strategies were only used when the opponent was detected to be always raise or always call. Otherwise, the agent employed an implicit modelling approach, combining a portfolio of the remaining five strategies in the same way as our case study. Further information about the strategies, such as abstraction details, are available in Appendix B.1.1.

Empirical Results. Similar to the results shown in our case study from the 2011 benchmark server, Table 6.2 shows the average bankroll of each of the 2012 competitors. Though our agent's ranking was a disappointing fourth, the results from this competition were interesting because the *magnitude* of every agent's bankroll was effectively an order of magnitude smaller than in past competitions. Moreover, as the full cross table of results in Table A.1 shows, this occurred across all pairs of agents and was not merely an artifact of computing an average bankroll.

There are a couple reasonable explanations for this outcome. First, the field of agents may have been exploitable, but no agent was able to tailor an exploitive response to its opponent during the competition. Alternatively, due to progressively better Nash equilibrium approximations, the field of agents may have been largely unexploitable. This possibility is supported by the participants' descriptions of their agents: many of whom indicated they used CFR or LP techniques to solve the game, or imitated prior strong agents. In fact, the top three agents all used CFR variants to solve the game. Whether it was due to the prevalence of Nash equilibrium approximation techniques, or insufficient incentives for competitors to deviate from equilibrium strategies, the total bankroll competition was showing symptoms that it was no longer eliciting agents capable of being modelled and exploited. In an attempt to address this, the total bankroll events were modified in subsequent years.

	AVG
slumbot	29 ± 3
little.rock	14 ± 3
zbot	13 ± 3
hyperborean.tbr	7 ± 2
patience	7 ± 3
neo.poker.lab	5 ± 3
entropy	4 ± 3
sartre	2 ± 3
huhuers	−8 ± 2
feste.tbr	−34 ± 4
little.ace	−38 ± 3

Table 6.2: Average winning rates from the 2012 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown.

Competition Changes. Following the 2012 ACPC, the total bankroll events were changed in two ways. First, to promote more exploitive agents, the total bankroll event in the heads-up limit Texas hold'em competition was seeded with exploitable "chump" agents similar to what was done in the RoShamBo Programming Competitions (Billings 1999). Including these chumps provides competitors with some incentive to deviate from an equilibrium strategy. However, unlike the RoShamBo competition, generating a variety of chumps that require distinct responses to exploit them is not trivial. Although each of the ACPC's total bankroll events may have benefited from chump agents, they were only incorporated into the heads-up limit competition due to the difficulty in constructing such agents.

Second, a limit was imposed on the possible winnings from any given agent. That is, for a given limit X, if in a match between agents A and B, A wins at a rate of $Y > X$, then A's winnings are capped at X and B's losses are capped at $-X$ when computing the total bankroll. The rationale for such a cap was to mitigate the impact of "kingmakers" on the competition's outcome and encourage agents that were capable of exploiting a wider range of opponents. Buggy agents (*e.g.*, POMPEIA in the 2011 heads-up no-limit competition) were a common source of such kingmakers in prior competitions and effectively decided the outcome of several total bankroll events. With the introduction of chumps, the cap also serves to reduce the risk of having a single chump decide a competition's outcome. The cap was chosen to be the value against an always fold strategy (*i.e.*, 750 mbb/g) because it provides a reference strategy that is simply described, trivial to employ, and generally regarded as being terrible. With more than two players, it is less obvious how to cap and redistribute winnings across multiple players. Although a cap of 750 mbb/g was used in the 3-player limit game, despite not being the value of an always fold strategy, it had no actual impact on the results. For other events, we will examine how this cap impacted the outcome of the competition. Our next results examine the first ACPC event including both capped winnings and chumps.

2013

Agent Design. The Hyperborean implicit modelling agent submitted in 2013 was similar to our agent from the 2012 competition. The portfolio consisted of four DBRs to the same agents as in the 2012 portfolio (*i.e.*, ASVP, RobotBot, TellBot, Tiltnet), except with a more aggressive value for each DBR's $P_{\max}$ parameter and different asymmetric abstractions. For further details about these DBR strategies, see Appendix B.1.2. This year's agent removed the Nash equilibrium approximation from the portfolio as the new portfolio performed similarly to the 2012 portfolio against a strong equilibrium-based agent (*i.e.*, the 2012 ACPC

	AVG
marv	200
feste_tbr	179
hyperborean_tbr	175
zbot	151
littlerock	144
propokertools	143
neo_poker_lab	143
HITSZ_CS_13	90
chump12	24
unamur_tbr	−60
liacc	−229
chump1	−391
slugathorus	−569

(a) Capped

	AVG
feste_tbr	210
marv	200
hyperborean_tbr	175
zbot	151
littlerock	144
propokertools	143
neo_poker_lab	143
HITSZ_CS_13	114
chump12	24
unamur_tbr	−60
liacc	−166
chump1	−391
slugathorus	−687

(b) Uncapped

Table 6.3: Average winning rates from the 2013 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g.

heads-up limit instant runoff Hyperborean agent). Furthermore, the special handling for always call and always raise opponents was removed as experiments found that our agent performed near or above the new winnings cap against such opponents regardless.

Empirical Results. We begin our discussion of the results by examining the general condition of the competition after the changes announced in 2012. Table 6.3a shows that the average bankrolls from the official results (now using the aforementioned cap of 750 mbb/g) returned to a similar magnitude as heads-up limit competitions prior to 2012. The full cross table of results, shown in Table A.2, suggests this is due both to the newly introduced chump agents along with more exploitable competitor agents (*i.e.*, HITSZ_CS_13, unamur_tbr, liacc, and slugathorus).

While these agents provided a more rewarding environment for exploitive agents, it is interesting to note that the weaker agents generally lost badly against equilibrium-based agents (*e.g.*, zbot and littlerock) even though a Nash equilibrium does not directly attempt to exploit such suboptimality. For example, the most unamur_tbr lost against any agent was 213 mbb/g. Against zbot, a CFR-based agent that tied for third place in the bankroll instant runoff event, it lost at a rate of 187. We can observe similar, though less extreme, results for the other exploitable agents. Slugathorus, for instance, lost to zbot at a rate of 449 while the most it lost by was 1,564 (which appears in the uncapped results shown in Table A.4). However, the competition's cap limits this to 750, leaving a much smaller margin of benefit for exploitive agents.

Unamur_tbr and slugathorus highlight challenges for both competitors and competition organizers alike in agent modelling tasks like the total bankroll events. For competition organizers aiming to elicit ideal agents that adapt to the other agents at hand, designing an evaluation mechanism is nontrivial. The evaluation should provide sufficient discriminatory power to distinguish between adaptive agents and stationary strategies. While seeding the competition with chump agents sought to provide this power, the incentives for adaptive agents *relative to an equilibrium strategy* often appeared relatively minimal in this competition. Admittedly, since each agent's true exploitability is unknown[4], competitors may have simply failed to discover and exploit vulnerabilities. However, even when vulnerabilities were exploited, as in slugathorus' case, the benefit of such exploitation was muted by the

[4] Although computing a fixed strategy's real game exploitability is feasible due to Johanson and colleagues (2011), the agents' strategies may not be fixed and are not publically available for evaluation.

	marv	zbot	neo_poker_lab	littlerock	propokertools	HITSZ_CS_13	liacc	slugathorus	AVG
hyperborean_tbr	−37 ± 8	−26 ± 9	−32 ± 10	−11 ± 9	−8 ± 9	159 ± 9	527 ± 21	617 ± 20	149
hyperborean_iro	15 ± 6	10 ± 4	−11 ± 6	22 ± 7	27 ± 8	157 ± 11	390 ± 15	441 ± 15	131

Table 6.4: Comparison of Hyperborean agents submitted to the instant runoff and total bankroll competitions of the 2013 ACPC heads-up limit Texas hold'em event. Results are in mbb/g with 95% confidence intervals shown.

competition's winnings cap. Designing an evaluation mechanism that strikes an appropriate balance between the disparate goals of eliciting agents that maximize utility – even if that utility comes from a single kingmaker – or agents that perform well against a broad range of opponents, such as equilibrium-based agents, is an ongoing challenge for the ACPC.

Competitors are also forced to proverbially thread this needle. As in the design of robust response strategies, being excessively exploitive may expose an agent to unnecessary risk against strong agents for little or no benefit. This is particularly relevant with the winnings cap, which artificially changes this exploitation-exploitability trade-off. The performance of feste_tbr illustrates this challenge. A less aggressive agent may have still hit the cap against both liacc and slugathorus, while potentially suffering smaller losses against the competition's stronger agents. Though the robust response techniques used to generate our portfolio have parameters to tune this aggression, uncertainty about the behaviour of other agents prior to the competition typically means such tuning is only done heuristically.

Next, we examine the performance of our implicit modelling agent, hyperborean_tbr. Table 6.3a shows that with the winnings cap, our agent had the third highest average bankroll behind marv (whose name was intended to be Bacalhau) and feste_tbr. To evaluate statistical significance, a bootstrap technique was used in place of the prior confidence interval based evaluation. The bootstrap technique resamples multiple sets of matches from the actual matches that were run. By evaluating the outcome (in this case, each agent's average bankroll) for each resampled "competition", we get a sense of the stability of the results. Specifically, one can answer how frequently agent A's average bankroll exceeds agent B's. The ACPC declares A's win statistically significant if this frequency is at least 95%. Statistical significance is evaluated in this manner for the remainder of our ACPC results. For example, Table A.3 shows the proportion of 1,000 resampled competition outcomes that the row player beat the column player. While marv beats feste_tbr with statistical significance, feste_tbr and hyperborean_tbr could not be separated with statistical significance.

Table 6.3b shows the competition's results if there was no winnings cap. Though four agents average bankrolls were impacted by the cap, it only affected the relative ranking of feste_tbr. Without the cap, feste_tbr would have placed first, followed by marv, and hyperborean_tbr, with each result being statistically significant (see Table A.5 for the bootstrap analysis). Examining the full uncapped results (Table A.4) highlights some shortcomings of our implicit modelling agent. Most notably, liacc and slugathorus were exploitable for considerably more than our agent managed. While this could be caused by slow adaptation or an overly passive portfolio of strategies, it may also be symptom that the portfolio is providing insufficient coverage over the space of opponent strategies.

Finally, we can also gain some insight about our agent by comparing it to hyperborean_iro, which took second place in the bankroll instant runoff event. Table 6.4 shows the performance of both agents against their mutual opponents. Note that the IRO and TBR agents from Feste and Unamur have been omitted since they did not participate in

	AVG
Escabeche	173
SmooCT	152
Hyperborean	143
Feste	141
Cleverpiggy	133
ProPokerTools	129
652	125
Slugathorus	123
Lucifer	8
TerribleTerrance	4
PokerStar	−89
HITSZ_CS_14	−201
chump9	−324
chump4	−515

(a) Capped

	AVG
Escabeche	174
Feste	168
Slugathorus	164
SmooCT	157
Hyperborean	153
Cleverpiggy	133
ProPokerTools	129
652	125
Lucifer	8
TerribleTerrance	4
PokerStar	−89
HITSZ_CS_14	−201
chump9	−342
chump4	−581

(b) Uncapped

Table 6.5: Average winning rates from the 2014 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g.

both events. Unsurprisingly, hyperborean_iro's Nash equilibrium approximation, which used the much larger abstract game described in Appendix B.1.2, performed marginally better against stronger agents. Despite this, our implicit modelling agent yields a 13.3% greater average bankroll by considerably outperforming hyperborean_iro against liacc and slugathorus. One potential solution for such vulnerability to stronger equilibrium-based agents is to increase the size of the abstraction used for the opponent's response in the portfolio's DBRs. However, as we showed in Figure 4.4, while this tends to improve a DBR's worst-case performance, it can lead to worse exploitation if our frequentist model is sparsely populated.

2014

Agent Design. Our 2014 implicit modelling agent, simply called Hyperborean (as no bankroll instant runoff event occurred this year), used a portfolio consisting of four strategies using asymmetric abstractions:

1–2. The data biased responses to Tiltnet and ASVP from the 2013 portfolio

3. A data biased response to data from Feste agents since 2011

4. An asymmetric abstract Nash equilibrium approximation designed to exploit mistakes made by equilibrium-based agents using smaller abstractions of the game.

Additional implementation details about these strategies are available in Appendix B.1.3.

Empirical Results. As before, Table 6.5 shows the capped and uncapped results from the 2014 competition. In the official capped evaluation, Hyperborean placed third behind Escabeche, the seafood-named successor to marv (Bacalhau) that took first place, and SmooCT. The ordering of these three agents was statistically significant, but Hyperborean's win over the fourth place Feste was not (beating it in only 89.4% of the bootstrap samples). Full cross tables of results are shown in Table A.6 along with bootstrapped statistical significance in Table A.7.

Unlike the 2013 results, the cap played a much larger role on the agent rankings. Without the cap, Feste moves from fourth to second, Slugathorus moves from seventh to third, while SmooCT and Hyperborean get pushed down into fourth and fifth, respectively. Though

Escabeche and Feste beat most agents beneath them with statistical significance, neither beat Slugathorus with significance (though Escabeche was very close, winning 93.3% of the time). SmooCT's win over Hyperborean also failed to reach the 95% threshold for statistical significance, winning in 82.3% of the bootstrap samples. For the complete bootstrap analysis, see Table A.9. The uncapped cross table of winnings in Table A.8 shows that all five of the above agents reached the cap against chump4, while Slugathorus was the only agent to exploit chump9 for more than the cap. Moreover, it shows that Hyperborean failed to exploit chump9 effectively: winning only 272 mbb/g on expectation compared to Slugathorus' 984 mbb/g. As in the 2013 competition, where we failed to exploit liacc or slugathorus as well as other agents, this may indicate that our portfolio provides insufficient coverage over the space of opponent strategies.

Marv Andersen's agents (including Calamari in the 2011 ACPC) have consistently managed to be successful in the total bankroll events by striking an effective balance between performance against strong agents and capitalizing on exploitable agents. Interestingly, his agents manage this despite being stationary strategies generated by training a neural net to imitate the play of previous ACPC winners (ACPC 2015, participants listings). Unfortunately, further details about his approach have not been published, making it is difficult to ascertain why this approach provides broad efficacy. In particular, if the neural nets have been trained on prior winners of the bankroll instant runoff events, which tend to be equilibrium-based strategies, then it is unclear why the resulting strategies tend to exploit opponents more effectively than the strategies they are based on.

While Marv's agents are certainly *exploitive* and can be viewed as using agent modelling since they aim to imitate prior agents, they also seem unsatisfying since they are not *adaptive*. If the scientific goal of the ACPC's total bankroll events is to elicit adaptive agent modelling techniques, then it is unclear how the competition can be designed to better promote such techniques. Although the lack of a clear direction on how to redesign the heads-up limit total bankroll event led to its indefinite discontinuation, the game remains an interesting domain for agent modelling research.

6.2.2 3-player Limit Texas Hold'em

The ACPC's 3-player limit Texas hold'em events offer competitors an interesting challenge: even if competitors could approximate a Nash equilibrium strategy, it is unclear how useful this solution concept would be since it only guarantees that no player can *unilaterally* deviate to improve their utility. In the ACPC's two-player poker variants, this provides a powerful worst-case performance guarantee. However, when multiple agents can collude together, a Nash equilibrium may not guard against, nor take advantage of, the possibility of collusion. While explicit collusion is typically prohibited in games like poker, agents may tacitly collude to their mutual benefit. For example, in the 3-player variant of Kuhn poker, Szafron and colleagues (2013) showed how one player, acting according to their part of an equilibrium profile, could transfer utility between the other two players. Through repeated games, two players could learn to cooperate with each other at the expense of the third.

Our ACPC results for 3-player limit Texas hold'em examine the performance of a static agent created using a DBR with asymmetric abstractions in the 2013 and 2014 total bankroll events. Part of the rationale for using a static agent was that the shorter 1,000 hand matches in the 2013 competition would provide even less time for an adaptive agent to learn. Though this means our results do not demonstrate an adaptive implicit modelling agent, as in our other ACPC results, it does highlight the potential value of using robust responses and asymmetric abstractions (investigated in Chapter 4) to create strategies in domains where collusion is possible. Note that although both of these years applied a winnings cap of 750 mbb/g to the results, it did not impact any of the results. As such, we only present the official capped results.

	AVG
hyperborean_tbr	296
littlerock	161
neo_poker_lab	117
HITSZ_CS_13	−95
kempfer	−177
liacc	−302

Capped/Uncapped

Table 6.6: Average winning rates from the 2013 ACPC 3-player limit Texas hold'em total bankroll competition. Results are in mbb/g.

	littlerock				neo_poker_lab			HITSZ_CS_13		kempfer	
	neo_poker_lab	HITSZ_CS_13	kempfer	liacc	HITSZ_CS_13	kempfer	liacc	kempfer	liacc	liacc	AVG
hyperborean_tbr	49 ± 19	282 ± 21	157 ± 26	338 ± 43	278 ± 27	176 ± 31	391 ± 41	385 ± 35	452 ± 47	455 ± 35	296
hyperborean_iro	49 ± 14	218 ± 28	143 ± 17	218 ± 37	234 ± 25	150 ± 19	260 ± 37	316 ± 30	302 ± 59	323 ± 53	221

Table 6.7: Comparison of Hyperborean agents submitted to the instant runoff and total bankroll competitions of the 2013 ACPC 3-player limit Texas hold'em event. Results are in mbb/g with 95% confidence intervals shown.

2013

Agent Design. The agent for the 2013 and 2014 total bankroll events used a data biased response to aggregate data of all ACPC competitors from the 2011 and 2012 3-player limit competitions. Asymmetric abstractions were used for the regret minimizing part of each player's strategy and the frequentist model used by data biased response. When constructing a DBR, recall that one must produce a strategy for each of the position in the game. In domains with more than two players, if we have different beliefs about the other players' behaviours then their relative order will impact the resulting response strategy. When training this DBR, we used the same abstractions and frequentist models for both opponents. For additional details about how this strategy was generated, see Appendix B.2.1.

Empirical Results. We begin our analysis of the 2013 ACPC results by examining the performance of our agent, hyperborean_tbr, in the total bankroll competition. Table 6.6 shows the average winnings of each agent. Our agent is a clear winner in this competition, achieving an expected winnings rate 84.3% greater than the second place finisher. The full results in Table A.10 show that these winnings come from across the field of agents, with hyperborean_tbr being the most profitable (on expectation) for any combination of agents. The entire ranking was statistically significant, with every agent beating all the agents ranked lower than them in 100% of the bootstrap samples (Table A.11).

Comparing the Hyperborean agents submitted in the total bankroll and bankroll instant runoff events reveals an interesting result about our DBR strategy. Richard Gibson (2014,

	AVG
Hyperborean_tbr	194
SmooCT	138
KEmpfer	92
HITSZ_CS_14	−150
Lucifer	−274

Capped/Uncapped

Table 6.8: Average winning rates from the 2014 ACPC 3-player limit Texas hold'em total bankroll competition. Results are in mbb/g.

	SmooCT		
	KEmpfer	HITSZ_CS_14	Lucifer
Hyperborean_tbr	76 ± 11	151 ± 8	214 ± 11
Hyperborean_iro	59 ± 6	134 ± 7	138 ± 11

	KEmpfer		HITSZ_CS_14	
	HITSZ_CS_14	Lucifer	Lucifer	AVG
Hyperborean_tbr	176 ± 9	240 ± 14	305 ± 10	194
Hyperborean_iro	143 ± 8	157 ± 10	229 ± 9	143

Table 6.9: Comparison of Hyperborean agents submitted to the instant runoff and total bankroll competitions of the 2014 ACPC 3-player limit Texas hold'em event. Results are in mbb/g with 95% confidence intervals shown.

Section 8.4) designed the "dynamic expert strategy" (Gibson and Szafron 2011; Gibson 2014, Section 7.2) used for the hyperborean_iro agent in both the 2013 and 2014 bankroll instant runoff events. His agent's strategy was generated using CFR-based techniques on an abstract game generated using two different granularities of card abstraction, both of which are considerably larger than the abstractions used by our DBR. Despite our agent's comparatively small size, Table 6.7 shows that it performs on par or better than hyperborean_iro against this field of opponents: yielding a 33.8% greater average bankroll. Most notably, our agent makes substantial gains in utility in any of the matches involving either HITSZ_CS_13 or liacc, almost all of which are statistically significant based on the 95% confidence intervals. This is likely due to the aggregate data of prior ACPC competitors providing more accurate beliefs about the behaviour of these agents, despite neither agent playing in past competitions. Note that although hyperborean_tbr performs at least as well as hyperborean_iro against this field, internal experiments pitting it against hyperborean_iro and the previous 2012 Hyperborean IRO agent showed hyperborean_iro winning and hyperborean_tbr losing at a small but statistically significant rate (10 ± 2 mbb/g).

2014

Although the increase in match length from 1,000 to 3,000 hands for the 2014 competition gave adaptive agents more time to learn, the 2013 Hyperborean agents were simply resubmitted for the 2014 competition. As such, we move directly to our analysis of the ACPC results.

Empirical Results. The results from the 2014 competition in Table 6.8 echo those of 2013. Hyperborean_tbr won the total bankroll competition in 2014 with an expected winnings rate 40.1% higher than the second place finisher. Table A.12 shows that these winnings come from the entire field again, with hyperborean_tbr having the greatest winning rate of the three players for any combination of agents. The rankings were statistically significant, with agents beating lower ranked opponents in 100% of the bootstrap samples (Table A.13). Finally, Table 6.9 shows us that hyperborean_tbr outperforms the much larger hyperborean_iro against the entire 2014 field, and particularly in matches with Lucifer (liacc's successor). For each pair of opponents, except SmooCT and KEmpfer, this improvement is statistically significant, with the 95% confidence intervals on the agents' expected winnings being disjoint. Overall, using hyperborean_tbr increased the average bankroll 35.2%.

6.2.3 Heads-up No-limit Texas Hold'em

Our final set of ACPC results showcases implicit modelling agents in the domain of heads-up no-limit Texas hold'em poker. As we return to a two-player zero-sum domain, recall that not only is CFR guaranteed to converge to a (abstract game) Nash equilibrium in this setting, but the worst-case performance guarantee provided by such a solution is also more meaningful since our opponent has no other agents to collude with. However, in contrast to the heads-up limit variant, the large range of bet sizes available to agents in no-limit makes the domain enormous and presents practitioners with new challenges. To combat the game's size, practitioners will abstract both the cards and the betting sequences: typically abstracting the raise action to a number of bets relative to the pot size. To generate high-quality heads-up no-limit strategies, practitioners commonly employ larger abstract games and, in turn, more computational resources than for heads-up limit.

Since practitioners may want to evaluate several candidate strategies when designing their portfolio, as in our heads-up limit case study, this makes building a portfolio of strategies a substantial computational burden. To illustrate, practitioners may generate candidates using robust responses to different agent models or by simply varying a confidence parameter like the $P_{\max}$ parameter for DBRs. Furthermore, building a full strategy profile for each of these robust responses requires practitioners to generate a strategy from each player's asymmetric point of view. Therefore, even attempting to tune the $P_{\max}$ parameter for a single agent model requires at least four abstract strategies to be generated. For the larger abstractions commonly used in heads-up no-limit, evaluating numerous candidates rapidly becomes impractical. As such, our no-limit portfolios were created without the same broad exploratory approach used in our heads-up limit case study.

Our ACPC results for heads-up no-limit Texas hold'em examine the performance of our implicit modelling agents against competitors from the 2013 and 2014 ACPC. Observe that although an agent's exploitability could be as bad as 200,000 mbb/g in the ACPC's heads-up no-limit domain, as opposed to 24,000 mbb/g in the limit variants, a strategy that always folds will still lose at a rate of 750 mbb/g. In our comparison of the capped and uncapped results, we note that the cap of 750 mbb/g had a substantial impact on most agents' average bankrolls and final ranking.

2013

Agent Design. Our implicit modelling agent for the 2013 ACPC total bankroll event uses a portfolio (detailed in Appendix B.3.1) of two abstract strategies:

1. A dynamic expert abstract Nash equilibrium approximation. This strategy was also used as part of the 2013 heads-up no-limit bankroll instant runoff Hyperborean agent.
2. A data biased response to aggregate data from all of the agents in the 2011 and 2012 heads-up no-limit ACPC events.

The portfolio's final design was motivated by several challenges that arise in the heads-up no-limit domain. As we alluded to earlier, the immense size of the heads-up no-limit domain amplifies many existing challenges in building a portfolio. To guard our DBRs against strong equilibrium-based strategies, such as those employed by at least three top agents in the 2012 ACPC, we chose to use a large abstraction that fully utilized the memory available on our computational resources. However, in addition to memory, CFR also tends to need more computational time to converge in a larger abstraction. With our available time and resources, it was impractical to generate fully converged strategies, never mind a variety of DBRs. Under these constraints, we chose to construct DBRs based on a single model: opting for an aggregate model of past agents as a similar approach yielded an effective DBR in our heads-up limit case study (*i.e.*, the Small-Static agent). Unfortunately, resource constraints also forced us to choose a limited selection of $P_{\max}$ values. Without prior results in no-limit to inform us on how this would impact the trade-off between worst-case and one-on-one performance, we chose values of 0.25 and 0.5.

To further guard against equilibrium-based opponents, we also prepared to use a large abstract equilibrium in the portfolio. While this may appear to be inconsequential to the design of the rest of the portfolio, the betting abstractions common in no-limit, which *remove* an agent's available actions, introduce a subtle challenge when constructing a portfolio. Namely, using different betting abstractions within the portfolio could impede information sharing during our off-policy importance sampling since strategies may take actions the other strategies are incapable of producing. Because of this, our DBRs were constructed using the same betting abstraction as the equilibrium approximation that was included in the final portfolio.

To construct our frequentist model, real game betting actions made by past ACPC agents were mapped into this common betting abstraction using hard (*i.e.*, deterministic) translation based on geometric similarity (Schnizlein 2009). Unfortunately, having multiple magnitudes of raise actions disperses our available observations throughout a much larger betting space, potentially causing much sparser frequentist models. To mitigate this concern, the DBR's frequentist model uses an asymmetric abstraction that ignores card information (*i.e.*, an abstraction with 1 bucket on each round) and only models agents on their abstract betting. One benefit of this approach is that it does not require any private information. However, as with prior explicit modelling techniques that discarded private card information (Rubin and Watson 2012b; Ganzfried and Sandholm 2011), this approach almost certainly conflates behaviours that would be valuable to differentiate.

Our final portfolio included only the DBR with a $P_{\max}$ parameter of 0.25. Although the more aggressive $P_{\max}$ of 0.5 yielded a substantially higher uncapped average bankroll in experiments on the 2012 ACPC benchmark server, it also performed poorly against equilibrium-based opponents. By comparison, the more passive DBR lost less against such opponents while still exploiting weaker opponents for more than the competition's winnings cap of 750 mbb/g.

For practitioners that are similarly limited in computational resources, an alternative approach to portfolio design, akin to the one taken in our heads-up limit case study, could allow a broader exploration of DBR candidates. Specifically, practitioners could generate and evaluate highly abstracted candidate DBRs against the DBR mimic strategies and identify high-value models through a submodular optimization. Information gathered during this initial phase could then be used to construct DBRs using finer granularity abstractions to guard against stronger opponents. However, this approach comes with caveats. First, increasing the granularity of the frequentist model's abstraction – either in an effort to better model the opponent or simply to share a common betting abstraction – would increase model sparsity. Moreover, as we observed in our asymmetric abstraction results, increasing the size of the opponent's abstraction tends to both improve worst-case performance and decrease one-on-one performance: with more substantial decreases for sparser models. While bootstrapping a portfolio in this manner may be beneficial, we do not explore it further.

Empirical Results. We begin our analysis of the 2013 heads-up no-limit total bankroll competition by examining the official capped results. Table 6.10 shows both the capped and uncapped average bankrolls. In the official capped results, our implicit modelling agent,

	AVG
slumbot	312
hyperborean_tbr	297
tartanian6	281
nyx	247
koypetition	237
neo_poker_lab	165
littlerock	150
entropy	45
Sartre	42
hugh	−58
kempfer	−449
liacc	−625
HITSZ_CS_13	−645

(a) Capped

	AVG
hyperborean_tbr	1474
neo_poker_lab	750
koypetition	528
tartanian6	476
slumbot	475
nyx	412
entropy	324
Sartre	315
hugh	259
littlerock	229
kempfer	−454
liacc	−1302
HITSZ_CS_13	−3486

(b) Uncapped

Table 6.10: Average winning rates from the 2013 ACPC heads-up no-limit Texas hold'em total bankroll competition. Results are in mbb/g.

hyperborean_tbr, has the second highest bankroll. However, the ranking of hyperborean_tbr was not statistically significant with respect to its neighbours: losing to slumbot (first place) in 85% of samples, and winning against tartanian6 (third place) 65% of the time. Complete bootstrapped statistical significance results are shown in Table A.15.

Examining the full cross table of results in Table A.14, we see that our implicit modelling agent suffered statistically significant losses against slumbot, tartanian6, nyx, koypetition, and littlerock. Similar to the heads-up limit results, these five agents all performed well in the bankroll instant runoff event, taking second through sixth place, respectively (with hyperborean_iro placing first). Of these five agents, four used a CFR-based technique to approximate an abstract equilibrium. However, hyperborean_tbr performs well against weaker agents: reaching the cap against five of the seven remaining total bankroll agents.

The uncapped average bankrolls show us that the cap had a dramatic effect on both the average bankrolls and the final rankings. Without the cap, hyperborean_tbr becomes the clear winner, with an average bankroll 96.6% greater than the runner-up, neo_poker_lab, who was promoted from sixth. Table A.16 reveals that this shift in results is largely due to several agents exceeding the cap when playing against kempfer, liacc, or HITSZ_CS_13. Moreover, hyperborean_tbr was also the only agent to exceed the cap against entropy or Sartre, winning at 1,157 and 974 mbb/g, respectively. At best, the other agents only managed expected winnings of 307 and 340 mbb/g. One possible explanation for this result against Sartre is that their agent has historically *imitated* agents from past competitions: producing the very behaviour our portfolio's DBR was designed to *exploit*.

Of all the agents, liacc is an exemplar of both the benefits and drawbacks of using a capped total bankroll. Every agent except HITSZ_CS_13 beat liacc by considerably more than the cap. Expected winning rates ranged from 1,689 to 6,111 mbb/g and hyperborean_tbr won at 5,709. Despite its otherwise poor performance, liacc is also the strongest agent against HITSZ_CS_13 with an expected winnings rate of 19,784 mbb/g. Without the cap, if liacc lost less substantially against other agents, HITSZ_CS_13 could have acted as a kingmaker for liacc. On the other hand, by capping the winnings at 750 mbb/g, liacc provides virtually no discriminatory power for identifying adaptive agents. Furthermore, the cap causes the counterproductive side effect of disincentivizing agents from maximizing utility. For instance, when designing hyperborean_tbr, we avoided our more aggressive DBR despite empirical evidence that it was the better choice for truly maximizing utility.

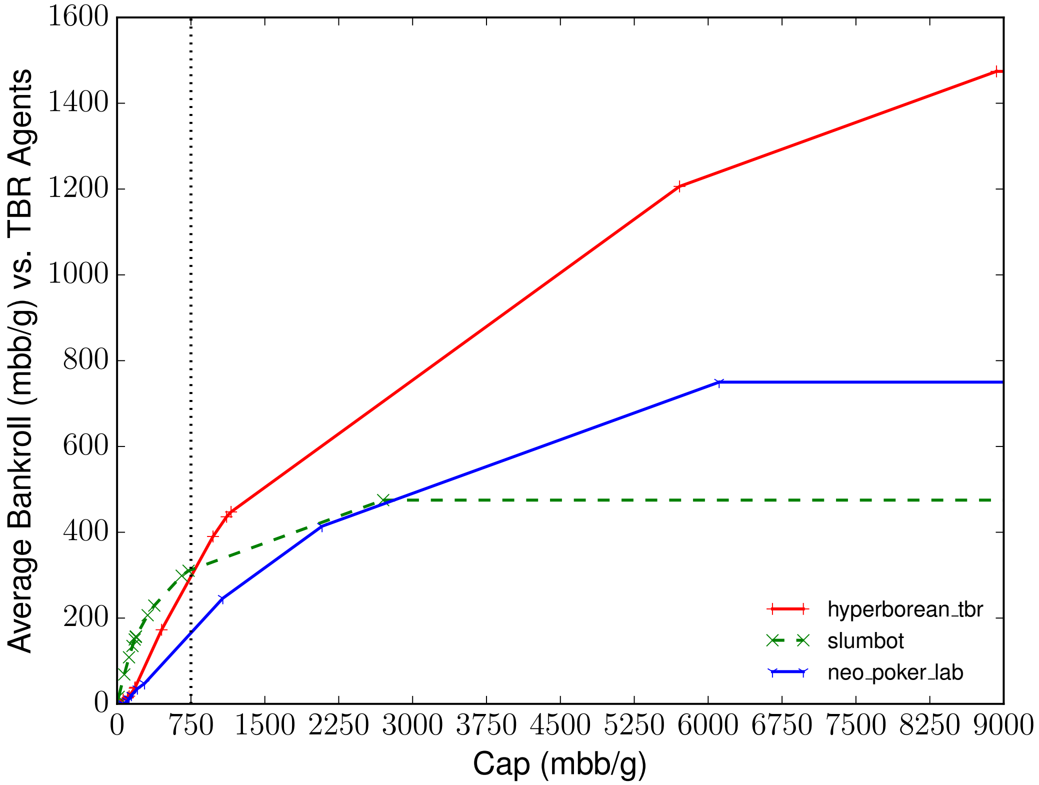


Figure 6.4: Impact of the expected winnings cap on the average bankroll of top agents from the 2013 heads-up no-limit ACPC. Solid versus dashed lines indicate an adaptive or static agent, respectively.

Even in less exceptional cases, the competition's capped evaluation regularly downplays or outright disregards substantive differences in performance against weaker agents. For example, against HITSZ_CS_13, slumbot and tartanian6 win at rates of 656 and 1,030 mbb/g, respectively, while hyperborean_tbr wins at 8,927 mbb/g. After the cap, hyperborean_tbr reaps little benefit over the stationary Nash equilibrium approximations, gaining a paltry 94 mbb/g over slumbot. In contrast, hyperborean_tbr's (capped) marginal gains over slumbot and tartanian6 are much greater when playing against Sartre or entropy, despite a dramatically smaller difference in true expected winnings. For instance, against Sartre, hyperborean_tbr gains 439 and 410 mbb/g over slumbot and tartanian6, respectively. Ideally, the competition's total bankroll evaluation mechanism would guard against kingmakers while consistently motivating agents to maximize utility.

We expand our analysis of how the cap impacts the ACPC's total bankroll evaluation by juxtaposing hyperborean_tbr with several close competitors under different values for the cap. Specifically, Figure 6.4 presents how the average bankrolls of hyperborean_tbr, slumbot, and neo_poker_lab vary depending on our choice of winnings cap. Each agent's performance curve is marked using a solid or dashed line based on if their description (ACPC 2015) specifies an adaptive or static agent, respectively. Each point on a curve corresponds to an expected winnings rate that the agent achieved against some opponent. An agent's average bankroll changes as a piecewise linear function of the cap that changes at each of these points. After the rightmost point on a curve, the cap exceeds the maximum expected winnings the agent achieved against the field of opponents; therefore, the agent's average bankroll will remain constant. For example, slumbot's average bankroll flattens at a cap of

	neo_poker_lab	koypetition	slumbot	tartanian6	nyx	entropy	Sartre	hugh	littlerock	kempfer	liacc	HITSZ_CS_13	CAP AVG	AVG
hyperborean_tbr	62 ± 99	−123 ± 90	−179 ± 95	−105 ± 91	−184 ± 81	1157 ± 132	974 ± 179	452 ± 57	−111 ± 84	1112 ± 165	5709 ± 384	8927 ± 565	297	1474
hyperborean_iro	210 ± 57	100 ± 62	20 ± 18	38 ± 77	67 ± 43	196 ± 58	177 ± 46	461 ± 52	133 ± 52	519 ± 57	3204 ± 197	542 ± 58	268	472

Table 6.11: Comparison of Hyperborean agents submitted to the instant runoff and total bankroll competitions of the 2013 ACPC heads-up no-limit Texas hold'em event. Results are in mbb/g with 95% confidence intervals shown.

2,701, which is the expected winnings it achieved against liacc. The total bankroll event's official results use a cap of 750 mbb/g, which is marked by the vertical dotted line, while uncapped results are specified by bankrolls at the figure's right edge.

Because hyperborean_tbr exploited more opponents to a greater degree, its average bankroll grows both more rapidly and for longer than either of the other agent's. Though hyperborean_tbr initially trails slumbot, the two agents are tied in average bankroll once the cap reaches 796 mbb/g. Once the cap reaches approximately 900 mbb/g, hyperborean_tbr would have a statistically significant victory over slumbot (based on 1,000 bootstrapped samples). Although hyperborean_tbr's average bankroll dominates neo_poker_lab's across the range of cap values, neo_poker_lab outperforms slumbot across a wide range of possible values. While the cap was intended to prevent a single kingmaker from deciding the total bankroll event, these results highlight how it can skew the outcome in favour of strong equilibrium-based agents like slumbot (which placed second in the instant runoff event). Since these agents obtain most of their utility by beating a wide range of opponents for comparatively small quantities, the cap has a relatively small impact on their performance.

Comparing the Hyperborean agents from the 2013 total bankroll and instant runoff events is interesting because hyperborean_iro also uses a portfolio of two strategies: the abstract equilibrium approximation used in hyperborean_tbr's portfolio, and an alternative strategy that both makes and better understands minimum-sized bets. However, hyperborean_iro plays the common equilibrium until the opponent is observed making a minimum-sized bet on at least 1% of the hands played so far. For further details about this agent, see Appendix B.3.1. Table 6.11 presents both Hyperborean agents' performance against the field of ACPC agents, including their average bankroll both with and without the cap. Much like other agents that fared well in the instant runoff event, such as slumbot or tartanian6, hyperborean_iro outperforms hyperborean_tbr against stronger equilibrium-based strategies while exploiting weaker opponents less. Overall, even though hyperborean_tbr suffered five statistically significant loses, whereas hyperborean_iro always won, our implicit modelling agent outperformed hyperborean_iro in terms of maximizing utility across the field of opponents: yielding a 10.8% and 312.2% increase in capped and uncapped average bankrolls, respectively. While each approach has its merits, these results highlight the dramatic improvements in utility that agent modelling approaches can provide.

Observe that these notably different results occur even with both Hyperborean agents incorporating a common equilibrium strategy. Unfortunately, since agent output was not captured during the ACPC, it is unclear how much each agent's strategies were played in these matches. Though this makes it difficult to attribute the differences in performance to particular strategies, internal experiments showed that hyperborean_tbr's DBR lost by more than 500 mbb/g against several abstract equilibrium strategies, including the Hyperborean entry from the 2011 ACPC bankroll instant runoff event. Since hyperborean_tbr acts according to a combination of every strategy in its portfolio, this DBR presents a likely cause of the stark contrast between the two agents. Note that in spite of the DBR's apparently poor performance against equilibria, the implicit modelling agent's losses were still less than 200 mbb/g on expectation, suggesting that the agent was able to adapt and use the portfolio's equilibrium strategy to guard itself against more substantial losses. The DBR's worst-case performance would likely be improved by lowering the $P_{\max}$ parameter further or potentially through using larger abstractions. While this could lead to improved performance in the capped average bankroll, our earlier evaluation on the 2012 ACPC benchmark server found that reducing the DBR's $P_{\max}$ to 0.25 from 0.5 resulted in substantial decreases in the uncapped average bankroll. In fact, if our goal was truly to maximize winnings across the field of ACPC agents, it would likely be better to increase $P_{\max}$.

2014

Agent Design. Our implicit modelling agent for the 2014 ACPC total bankroll event introduces a third strategy to the previous portfolio:

1. The Nash equilibrium strategy from the 2013 portfolio.
2. The DBR from the 2013 portfolio, run for additional iterations of CFR.
3. A data biased response based on data from agents that were not beaten by the 2013 Hyperborean TBR entry for at least 750 mbb/g (using the same asymmetric abstractions and a $P_{\max}$ parameter of 0.25).

Our intent behind this new DBR was to provide better coverage against strategies that our previous portfolio failed to fully exploit to the value of the cap. As with our previous DBR, we limited the $P_{\max}$ parameter to 0.25 to mitigate our risk against the competition's many equilibrium-based agents.

Empirical Results. Table 6.12 shows the average bankrolls for our final ACPC results from the 2014 total bankroll event. Unfortunately, Hyperborean_tbr placed poorly according to the capped bankrolls, ending up in seventh place with statistical significance (see Table A.19 for full bootstrap details).

All of the six agents ranked above Hyperborean_tbr in the (capped) total bankroll used a stationary strategy based on their descriptions (ACPC 2015). With the exception of Feste_tbr, each of these agents used a CFR-based algorithm to approximate an abstract equilibrium and were ranked in the top six places of the instant runoff event (with Hyperborean_iro placing third). Both Feste_tbr and Feste_iro, which placed seventh in the instant runoff event, used linear programming techniques to generate their strategies. While Feste_iro approximated an abstract game equilibrium, Feste_tbr used a “slightly more aggressive” strategy that was constructed by solving for a restricted Nash response against a uniform random opponent (based on their 2013 heads-up limit agent description). However, unlike the heads-up limit Feste agents, these agents used a single strategy and did not attempt to adapt online.

Dissecting the full results in Table A.18, we observe outcomes consistent with the 2013 ACPC heads-up no-limit results. First, our implicit modelling agent suffers moderate losses against these highly ranked instant runoff agents while performing very well against the four lowest ranked agents (Rembrant3, KEmpfer, HITSZ_CS_14, and Lucifer). Hyperborean_tbr was one of only two agents that capped all four agents, which were the only agents beaten by more than the cap. However, of the top six agents, even the agent that fared the worst against the bottom four (HibiscusBiscuit) still won 90.8% of the possible capped

	AVG
Tartanian7	342
Nyx	309
Prelude	306
Slumbot	304
HibiscusBiscuit	241
Feste_tbr	198
Hyperborean_tbr	162
LittleRock	143
SartreNLExp	130
PijaiBot	7
Rembrant3	−424
KEmpfer	−454
HITSZ_CS_14	−629
Lucifer	−635

(a) Capped

	AVG
Hyperborean_tbr	944
SartreNLExp	693
Nyx	597
Tartanian7	498
Slumbot	470
Prelude	470
HibiscusBiscuit	365
Feste_tbr	359
PijaiBot	326
LittleRock	202
KEmpfer	−252
Rembrant3	−416
HITSZ_CS_14	−1801
Lucifer	−2455

(b) Uncapped

Table 6.12: Average winning rates from the 2014 ACPC heads-up no-limit Texas hold'em total bankroll competition. Results are in mbb/g.

utility; Tartanian7, who won both the IRO and TBR events, managed 92.2%. This leaves a slim margin for adaptive agents to improve over equilibrium-based strategies when playing against such highly exploitable opponents.

Without the cap, we again see a dramatic shift in both the average bankrolls and the final rankings. Hyperborean_tbr moves from seventh place to a statistically significant first place finish. SartreNLExp, which is also an adaptive agent, makes a similarly large jump from ninth to second place. In turn, this demotes the six top (stationary) agents, with the former winner, Tartanian7, dropping from first to fourth. Full significance results for the uncapped bankroll are presented in Table A.21. Against this field of agents, Hyperborean_tbr's uncapped average bankroll is 36.2% and 89.5% greater than those of SartreNLExp and Tartanian7, respectively. The cross table of uncapped winnings in Table A.20 shows that Hyperborean_tbr's improvement is largely due to consistently outperforming other agents in matches against the four capped agents: securing the highest expected winnings versus KEmpfer and HITSZ_CS_14, and the second highest winnings versus Rembrant3 and Lucifer (though these rankings were not always statistically significant). Interestingly, this is despite the fact that Hyperborean_tbr did not use the data from the predecessors of KEmpfer, HITSZ_CS_14, and Lucifer (previously liacc) since Hyperborean_tbr also capped them in the 2013 heads-up no-limit competition. The fact that Hyperborean_tbr largely outperforms other agents against these capped agents may suggest, disappointingly, that competitors are not capitalizing on past data to tailor their agents to opponents even on a year-to-year basis.

The magnitude of the cap has a slightly more nuanced impact on the outcome of the 2014 competition as multiple agents were in close contention for first place at different cap values. Figure 6.5 shows the effect of the cap's magnitude on the average bankrolls of four top agents: Hyperborean_tbr, Tartanian7, SartreNLExp, and Nyx, who ranked second in total bankroll and and third in uncapped total bankroll. The results echo those of 2013, with static equilibrium-based strategies securing most of their potential utility at comparatively small cap values and levelling off relatively quickly. Although Hyperborean_tbr outperforms SartreNLExp prior to the official cap of 750 mbb/g, it takes until a cap of 2,035 mbb/g to match Tartanian7, 2,180 mbb/g to tie with Nyx, and approximately 2,500 mbb/g to win the competition with statistical significance.

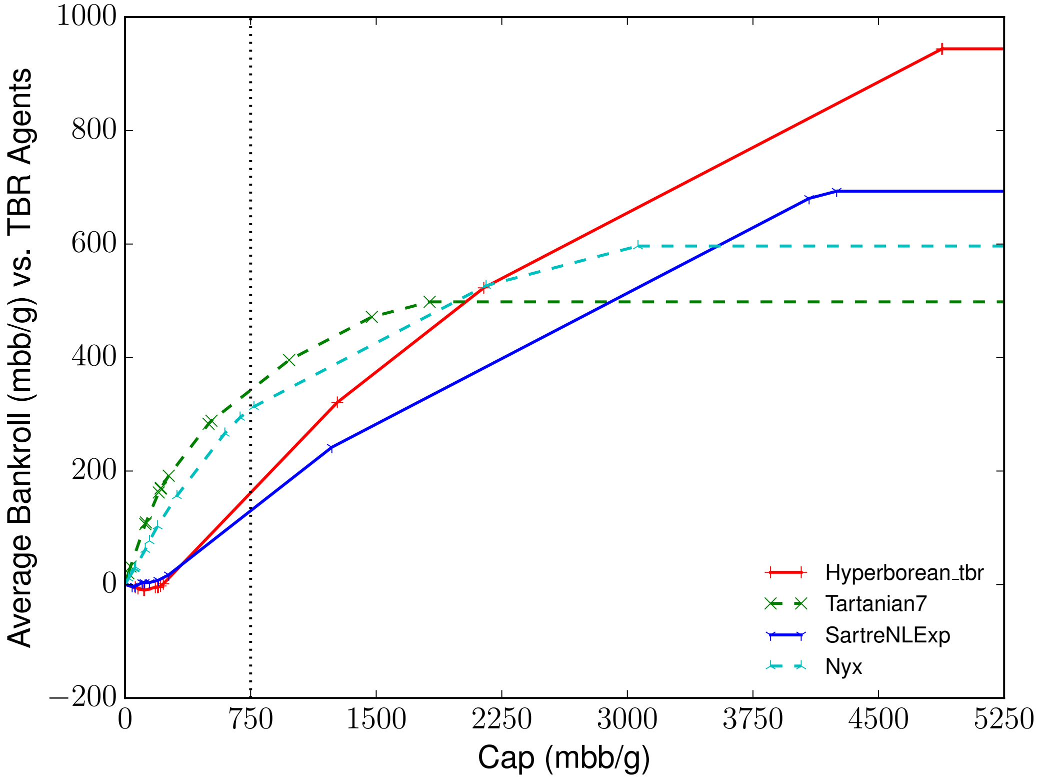


Figure 6.5: Impact of the expected winnings cap on the average bankroll of top agents from the 2014 heads-up no-limit ACPC. Solid versus dashed lines indicate an adaptive or static agent, respectively.

	SartreNLExp	Nyx	Tartanian7	Prelude	Slumbot	HibiscusBiscuit	PijaiBot	LittleRock	KEmpfer	Rembrant3	HITSZ_CS_14	Lucifer	CAP AVG	AVG
Hyperborean_tbr	−117 ± 106	−194 ± 136	−212 ± 113	−229 ± 117	−199 ± 75	−112 ± 100	181 ± 93	61 ± 67	2142 ± 257	1268 ± 123	4883 ± 986	4877 ± 696	181.56	1029.09
Hyperborean_iro	110 ± 41	54 ± 44	−21 ± 16	−10 ± 15	16 ± 14	40 ± 55	544 ± 86	297 ± 57	661 ± 71	926 ± 60	1293 ± 171	2311 ± 123	328.46	518.54

Table 6.13: Comparison of Hyperborean agents submitted to the instant runoff and total bankroll competitions of the 2014 ACPC heads-up no-limit Texas hold'em event. Results are in mbb/g with 95% confidence intervals shown.

Our final evaluation compares the Hyperborean_tbr and Hyperborean_iro agents. Similar to prior instant runoff agents, Hyperborean_iro approximates a Nash equilibrium in a large abstract game. However, in this case, the abstract game used an asymmetric betting abstraction, which provided more betting options for our opponent than our own agent (details available in Appendix B.3.2). This choice was motivated by our asymmetric abstraction finding that using a finer granularity abstraction for your opponent tends to yield strategies that are less exploitable in the worst case. Table 6.13 presents the agents' performance against their mutual opponents. Note that Feste_tbr is not included in the comparison, as Hyperborean_iro did not play agents exclusively in the total bankroll event. Unsurprisingly, Hyperborean_iro fares better than Hyperborean_tbr against the top (capped) total bankroll agents, as these agents all employ abstract equilibrium approximations. Interestingly, Hyperborean_iro and many top equilibrium-based agents managed to exploit PijaiBot and LittleRock. This opportunity for exploitation was missed by Hyperborean_tbr, and could be a symptom of insufficiently covering the space of opponent strategies. Unfortunately, though Hyperborean_tbr substantially outperforms Hyperborean_iro against the four lowest ranked agents, it provides very little benefit when utilities are capped. In fact, we would have been better off submitting Hyperborean_iro to the total bankroll event, as Hyperborean_tbr yields a 44.7% *decrease* in capped average bankroll. This margin would likely be even worse if results against Feste_tbr were included. However, when considering uncapped bankrolls, Hyperborean_tbr is clearly better: winning 98.5% more on average.

6.3 Summary

We presented a step-by-step approach of how to build adaptive implicit modelling agents for complex human-scale problems using a synthesis of techniques from several areas including robust responses in extensive-form games, submodular optimization, variance reduction, multi-armed bandits, and new contributions from this dissertation. Using several variants of Texas hold'em poker, we validate our implicit modelling approach through implementations of implicit modelling agents built using public data from past Annual Computer Poker Competitions. In our heads-up limit Texas hold'em case study, we show that implicit modelling agents can not only outperform other baseline response techniques, but also would have won the 2011 Annual Computer Poker Competition's total bankroll event.

We also implemented seven agents using techniques from this dissertation that were submitted to the ACPC's total bankroll events between 2012 and 2014. In 3-player limit Texas hold'em, our agents combined data biased responses with the asymmetric abstractions we investigated earlier in this dissertation. These agents not only won the competitions, but also substantially outperformed more sophisticated CFR-based agents against the entire field of opponents. Of the five implicit modelling agents that we implemented for the heads-up limit and heads-up no-limit Texas hold'em competitions, three agents achieved a top-three ranking. Furthermore, in terms of raw utility maximization in heads-up no-limit, our implicit modelling agents dramatically outperformed every other agent.

Combined with the ACPC's focus on real-time interaction and short matches that demand quick adaptation, these poker games present a sophisticated human-scale agent modelling task that embodies the gamut of challenges facing agent modelling practitioners. While traditional explicit modelling approaches struggle with the practical challenges of such complex human-scale domains, our results demonstrate that implicit modelling is a viable and promising alternative for agent modelling in these domains. Moreover, our implementations provide agent modelling practitioners with a foundation for developing their own adaptive implicit modelling agents. However, there remains several potential avenues to continue refining the implicit modelling framework, which we discuss in the next and final chapter before concluding the dissertation.

Chapter 7

Conclusions and Future Work

In this dissertation, we addressed a fundamental question in the field of agent modelling.

> In complex multiagent domains, how can an agent efficiently tailor its behaviour to other agents during interaction in order to maximize its performance?

We examined this question using a variety of poker domains that embody the spectrum of challenges facing agent modelling practitioners, including: very large state spaces, a limited quantity of stochastic observations, imperfect information, potentially dynamic behaviour from other agents, and an innate objective to maximize performance. This final chapter concludes the dissertation by reviewing the contributions we have made to the field of agent modelling through our exploration of this question. Furthermore, we discuss several potential avenues for future research on this topic.

7.1 Contributions

This dissertation consisted of three main contributions, each of which aimed to address distinct aspects of this essential agent modelling task.

Implicit agent models. In Chapter 3, we characterized and contrasted two approaches to agent modelling: the traditional approach, in which an agent responds to a generative model of agent behaviour learned during interaction, and an alternative approach that we call implicit modelling. In the implicit modelling framework, practitioners use online observations to estimate the utility of a portfolio of strategies. By modelling other agents solely through their impact on the portfolio's utility, implicit modelling alleviates many challenges that explicit models face in complex domains like poker. However, this approach also raises questions of how a practitioner should build and subsequently use the portfolio. We explored how to construct a portfolio prior to interaction. This approach both affords practitioners superior computational resources that can be exploited to build robust response strategies and helps ensure the agent can act in real-time during interaction.

We demonstrated the viability of this approach in complex human-scale domains through detailed implementations of implicit modelling agents in the variants of Texas hold'em poker played in the Annual Computer Poker Competition. We evaluated our implicit modelling agents in Chapter 6 by means of a thorough case study and analysis of results from prior ACPC events involving our agents. Our case study demonstrated that these agents outperform several baseline approaches in terms of maximizing utility. Our examination of the ACPC events presented a novel analysis of the ACPC's total bankroll agent evaluation mechanism. Furthermore, our agents performed particularly well in past heads-up no-limit Texas hold'em ACPC events, where they obtained substantially greater raw utility than other agents. In both cases, our agents were more profitable than conventional strategies based on Nash equilibrium approximations.

Asymmetric abstractions. In multiagent domains, an agent's beliefs about how other agents will or could act plays a significant role in their own behaviour and, in turn, their performance. For complex domains, some form of state-space abstraction is typically necessary to make the problem tractable, forcing practitioners to divide limited computational resources between the representation of each agent's behaviour. Although it is common to assume symmetric representations – where all agents distinguish states in the same way – we showed in Chapter 4 that this default belief does not optimize an agent's worst-case performance or its performance against suboptimal agents.

Our experiments provided the first empirical analysis of asymmetric abstractions in a complex human-scale domain, addressing both abstract Nash equilibria and robust responses in heads-up limit Texas hold'em. Our results reinforced prior observations of a trade-off between worst-case and one-on-one performance (Johanson et al. 2012b), extending the trend to cases where both players are abstracted, and discovered the first abstraction pathologies outside of a toy domain. In the robust response setting, we presented the first evaluation of restricted Nash responses (Johanson, Zinkevich, and Bowling 2008) and data biased responses (Johanson and Bowling 2009) in terms of their worst-case performance in the unabstracted game. We showed that not only does a similar performance trade-off exist for robust responses, but the quantity of one-on-one performance sacrificed depends on the accuracy of the agent model: with robust responses to accurate models, such as RNRs, producing responses that dominate their symmetric abstraction counterparts in terms of both performance measures. Furthermore, we demonstrated that robust responses, and particularly DBRs, can be less exploitable than Nash equilibrium approximations using similar abstractions. Finally, we found that for robust responses based on observations, such as DBRs, tailoring the abstraction used for the agent model to the quantity of available data can yield higher one-on-one performance. These asymmetric abstraction findings guided the portfolio design for all of our aforementioned implicit modelling agents.

Decision-theoretic clustering. To maximize utility in multiagent domains, agents generally need to condition their behaviour on how other agents in the environment act. Although customizing a response to each situation would be ideal, this is often impractical and agents are limited in their ability to personalize responses. Faced with such limitations, which responses should an agent employ to best maximize utility? Practitioners must confront this question when constructing an implicit modelling agent's portfolio prior to interaction: maximizing coverage over the space of potential agent behaviours while limiting the portfolio's size to avoid hampering its use online.

Our final contribution, presented in Chapter 5, addressed this problem by investigating how to elicit groups of similar agent behaviours. Traditional spatial clustering techniques, such as k-means, could be used to group agents together based on a distance metric. However, we illustrated that these techniques can fail to capture similarity in how an agent should respond to the resulting clusters to maximize utility. Segmentation problems previously introduced and formalized this decision-theoretic clustering task. Unfortunately, existing techniques for segmentation problems are either inapplicable or infeasible for extensive-form games with large response spaces, like our poker domains. We demonstrated that certain domains have structure that can be exploited to allow for an efficient greedy approximation algorithm for this problem. We proved worst-case approximation bounds on the quality of solutions produced by our algorithm. Finally, we demonstrated how to apply this algorithm to extensive-form games, and empirically demonstrated the value of this approach by comparing it to k-means for clustering agent behaviours in two toy games of poker.

7.2 Directions for Future Work

This dissertation opens up new avenues for further research in agent modelling and, more broadly, the range of complementary fields involved in our implicit modelling framework. Our effective demonstration of the implicit modelling framework in complex human-scale domains extends the reach of agent modelling techniques to sophisticated applications. In addition to such potential applications, we pose unresolved questions that arose throughout this research. We examine five general directions for potential research.

7.2.1 Moving Beyond Poker

Our validation of the implicit modelling framework largely focussed on empirical results in two-player zero-sum poker domains. However, since the techniques we employed do not require domains with these characteristics, they should be broadly applicable. This claim is supported by the successful application of similar approaches to domains outside of poker, such as the cooperative ad hoc teamwork task examined by Barrett and Stone (2015). Regardless, demonstrating these techniques in non-poker domains would provide valuable verification that they are, in fact, portable. For example, the results from our investigation of asymmetric abstractions suggested that being more pessimistic about the opponent's abstraction (*i.e.*, giving them a finer granularity abstraction) resulted in better worst-case performance. Does this finding translate into a cooperative domain, where being pessimistic would mean assuming a *coarse* abstraction for an ally? Furthermore, what general properties of implicit modelling be proven theoretically? Is there a useful theoretical relationship between implicit and explicit modelling?

7.2.2 Efficient Strategy Evaluation

Evaluating an agent's performance from a limited quantity of observations is a perennial challenge for designers of autonomous agents. Though variance reduction techniques for agent evaluation have been previously explored (Kocsis and Szepesvári 2005; Zinkevich et al. 2006; Billings and Kan 2006; Bowling et al. 2008; White 2010; Davidson 2014), they continue to be an important research topic for high variance domains like poker.

Online Variance Reduction. For the implicit modelling framework, efficiently estimating the utility of a portfolio's strategies during online interaction is vital for rapid adaptation. Although our implementations of implicit modelling agents used utility estimators augmented by importance sampling and imaginary observations, further exploration of existing and alternative variance reduction techniques could produce more effective implicit modelling agents. However, note that some existing techniques are focussed on post hoc variance reduction and rely on perfect information observations: making them unsuitable for evaluating agents online in an imperfect information domain like poker. Aside from imaginary observations, existing techniques capable of producing unbiased estimators in such settings – *e.g.*, LExp (Kocsis and Szepesvári 2005), MIVAT (White 2010, Section 5.3.5), and data baseline (Davidson 2014) – have yet to be applied to implicit modelling agents. Furthermore, any potential synergy between these techniques has not been investigated.

Off-policy Variance Reduction. Another desiderata for a portfolio's utility estimators is that they are efficient despite off-policy observations. When using importance sampling to update an estimator from off-policy observations, the estimator's variance can become large when rare outcomes are observed. While techniques like LExp and imaginary observations can help mitigate the impact of rare events, they assume existing strategies. Instead, can practitioners exploit knowledge of how they will estimate utility online during portfolio construction to produce "compatible strategies" that reduce estimator variance? Furthermore, can exploration strategies be designed to actively gather salient information that improves estimator efficiency?

7.2.3 Online Adaptation

The task of balancing between exploration and exploitation is a recurring challenge in machine learning. In our implicit modelling framework, we focused on multi-armed bandit techniques for combining the agent's portfolio and addressing this trade-off. While these techniques can provide theoretical guarantees on an agent's regret, the guarantees vary in several notable characteristics. For instance, Exp3G bounds the expected regret relative to the best strategy in hindsight assuming an oblivious adversary. However, alternative techniques can provide guarantees that assume different dynamics in the reward process (*e.g.*, adversaries that are stationary, oblivious, or themselves adaptive), evaluate regret relative to a changing strategy (*e.g.*, shifting/tracking regret), or bound regret with high probability instead of only in expectation.

Ideally, an agent should seek to maximize its performance regardless of the type of agents it is interacting with. In this sense, our choice of Exp3G may not be ideal since stationary strategies are commonplace for computer agents (*e.g.*, equilibrium strategies) and humans are likely to adapt their behaviour based on past observations. Furthermore, since Exp3G only bounds the *expected* regret, it may fail to consistently perform well. Such inconsistent performance may be particularly undesirable when an agent will have a small number of interactions, which is often the case when interacting with humans. Practitioners attempting to tailor an agent to the disparate natures of human and computer behaviour would benefit from additional guidance on the practical consequences of different online learning techniques in these settings. Several existing alternatives to Exp3G warrant particular consideration for our implicit modelling framework, including: Abernethy and Rakhlin's (2009) work on adaptive bandits, McMahan and Streeter's (2009) technique for improving the bounds of bandit problems with a small number of experts, and several of the refinements discussed by Bubeck and Cesa-Bianchi (2012, Section 3.4).

Another avenue to consider for adapting an agent's behaviour is to explicitly model the dynamics of the other agents. For instance, in prior work we examined how to use Bayesian state estimation techniques to explicitly model dynamic agents in Kuhn poker (Bard and Bowling 2007; Bard 2008). Like other explicit modelling techniques, scaling this approach to human-scale domains presented considerable practical challenges. However, it may be feasible to apply state estimation techniques in the implicit modelling framework's low-dimensional space of the portfolio's utilities.

7.2.4 Robust Responses

Responding to beliefs about the behaviour of other agents is a fundamental task for agent modelling practitioners. While responding in a manner that maximizes utility has been a de facto approach to this problem, Johanson and colleagues' (2008) showed that this can produce brittle responses that generalize poorly and perform badly in the worst case. Robust response techniques that accommodate for uncertain models of agent behaviour have been essential to constructing effective responses for our portfolios. As such, improving these techniques is a clear path to stronger implicit modelling agents and more effective agent modelling.

Real Game Robustness. When generating robust responses in an abstract game, which is the standard approach to make human-scale domains practical, the responses may not be robust in the original game. This problem is analogous to how abstract equilibrium strategies only guarantee optimality within their abstract game. Johanson and colleagues (2012b) addressed this problem for abstract equilibria with their CFR-BR algorithm by training against an unabstracted best response. Combining this approach with a restricted Nash response would yield a technique for generating robust responses that optimally trade off worst-case performance in the real game and one-on-one performance against a model. Furthermore, this may provide a mechanism to address CFR-BR's tendency to perform poorly one-on-one. Evaluating how this approach would impact other robust response techniques, such as data biased responses, is an open question. However, our earlier examination of using asymmetric abstractions for robust responses (Section 4.2.2) provides preliminary evidence supporting this approach, particularly with relatively accurate agent models.

Generative Model Efficiency. To build responses, practitioners typically must estimate a generative model of an agent's behaviour from limited observations. In complex domains, efficiently estimating an agent's behaviour in each unique situation is impractical and many situations will be poorly sampled. Furthermore, when modelling humans, their limited interaction speed typically exacerbates such estimator efficiency problems. How can generative models be estimated efficiently using less data? Can generalization techniques mitigate model sparsity while preserving model accuracy? Although the abstraction techniques used in our frequentist models for generalization produced competitive responses, our current approach has two notable limitations.

First, each observation contributed to only a single information set in the frequentist model's abstract game. This hard mapping limits generalization by artificially separating observations that could contribute information about other abstract information sets. For instance, since our frequentist models of heads-up and 3-player limit agents only used card abstraction, observations were not generalized across different player betting sequences. Prior work on generalization in poker has approached this problem by defining a softer notion of similarity between information sets (Schnizlein, Bowling, and Szafron 2009; Rubin and Watson 2012a; Schauenberg 2006, Section 4.3). However, we are unaware of any investigation into the impact that different generalization techniques have on not only the efficiency and accuracy of a generative model, but also the quality of its robust response.

Second, common poker abstraction techniques have focussed on producing strong abstract equilibria strategies rather than distinguishing behaviours of suboptimal agents. One illustration of this is the common assumption of card isomorphisms: where situations that differ only by a permutation of the cards' suits are not distinguished since an equilibrium strategy would treat them identically. However, generative models based on this assumption may discard valuable information for responding to suboptimal agents. The distinction between these objectives raises questions regarding how to estimate a generative model. Can the relevant information for maximizing a response's performance be compactly represented and efficiently estimated? Can an agent estimate relevant information more efficiently by prioritizing and actively learning about the most valuable information?

Extending Decision-theoretic Clustering. The decision-theoretic clustering technique we introduced in Chapter 5 can be used to inform practitioners about which responses are valuable to include in a portfolio. However, to apply the technique in practice, two challenges will likely need to be addressed. First, practitioners are typically limited to *observations* of an agent's strategy as opposed to knowing their complete strategy. While observations could be used to estimate a generative model for the strategy, we have not investigated how this impacts clustering quality. Second, in human-scale domains the cost of computing (a quadratic number of) best responses is prohibitive. Though recent accelerated best response techniques (Johanson et al. 2011; Johanson et al. 2012b) and parallelization of our algorithm can help overcome this problem, it may be preferable to sacrifice clustering quality for improved efficiency. How can practitioners adapt our technique to address these limitations?

One potential solution for both of these problems may stem from the portfolio pruning we performed in Section 6.1.1. Instead of approaching the segmentation problem as a partitioning optimization, our pruning technique employed a proxy for the maximum coverage optimization. Observations of each agent were used to generate a DBR and its corresponding mimic of the agent. Formulating the maximum coverage optimization in terms of these strategies both allows for the use of observations and only requires a linear number of response computations. In addition to the positive results we obtained from pruning in our case study, preliminary experiments that clustered Leduc hold'em strategies through a similar maximum coverage optimization yielded comparable performance to our algorithm. This suggests that such an approach may avoid the aforementioned limitations and allow for practical clustering in human-scale domains. The performance of such alternative approaches and the impact of aggregating observations within the resulting clusters remains unexplored.

7.2.5 Evaluating Adaptive Agents

Our final direction for further research is how to design mechanisms for evaluating adaptive agents. Unlike Nash equilibrium approximations, which have an invariant and clear (albeit potentially difficult to measure) objective, there is no general way to evaluate how well an adaptive agent approximates ideal behaviour. Furthermore, an agent's performance with respect to a small sample of agents, such as a competition's participants, may not be indicative of their performance against the broader agent population. This presents competition organizers and agent designers with the task of choosing an evaluation mechanism. As our earlier analysis of past Annual Computer Poker Competitions demonstrates, this choice can have a substantial impact on the outcome. What are the desiderata of a mechanism seeking to elicit effective adaptive agents? How can the mechanism account for outliers in the sampled agents while adhering to the objective of maximizing utility? Investigating these questions and potential mechanisms for evaluation would provide valuable guidance for this problem.

7.3 Summary

This dissertation contributes several techniques for tailoring an agent to the behaviour of other agents. The foundation of these techniques is the implicit modelling framework for agent modelling. We characterize this framework and demonstrate its efficacy and efficiency in complex human-scale poker domains where traditional agent modelling approaches have generally been unsuccessful. We focus on poker domains not only because they better reflect real-world challenges, as von Neumann famously observed, but also because poker presents an ideal problem for exploring the gamut of challenges facing agent modelling practitioners. Despite this focus, the versatility of our techniques and observations suggests that our contributions can be broadly applied: providing a practical approach for real-time online agent modelling in complex multiagent domains.

Bibliography

Abernethy, Jacob and Alexander Rakhlin (2009). “Beating the Adaptive Bandit with High Probability”. In: *Proceedings of the 22nd Conference on Learning Theory.* COLT 2009 (cit. on p. 94).

Abou Risk, Nick and Duane Szafron (2010). “Using Counterfactual Regret Minimization to Create Competitive Multiplayer Poker Agents”. In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems.* AAMAS 2010, pp. 159–166 (cit. on p. 13).

ACPC (2015). *The Annual Computer Poker Competition.* `www.computerpokercompetition.org`. [Online; accessed 25-November-2015] (cit. on pp. 9, 31, 44, 66, 79, 85, 87).

Argall, Brenna, Sonia Chernova, Manuela M. Veloso, and Brett Browning (2009). “A survey of robot learning from demonstration”. In: *Robotics and Autonomous Systems* 57.5, pp. 469–483 (cit. on p. 31).

Arthur, David and Sergei Vassilvitskii (2007). “k-means++: The Advantages of Careful Seeding”. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms.* SODA 2007, pp. 1027–1035 (cit. on p. 61).

Auer, Peter, Nicolò Cesa-Bianchi, and Paul Fischer (2002). “Finite-time Analysis of the Multiarmed Bandit Problem”. In: *Machine Learning* 47.2-3, pp. 235–256 (cit. on p. 22).

Auer, Peter, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire (1995). “Gambling in a rigged casino: The adversarial multi-armed bandit problem”. In: *Proceedings of the 36th Annual Symposium on Foundations of Computer Science.* FOCS 1995, pp. 322–331 (cit. on pp. 22, 23, 36).

Auer, Peter, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire (2002). “The nonstochastic multiarmed bandit problem”. In: *SIAM Journal on Computing* 32.1, pp. 48–77 (cit. on pp. 22, 23).

Bard, Nolan (2008). “Using State Estimation for Dynamic Agent Modelling”. MSc. Thesis. University of Alberta (cit. on pp. 29, 94).

Bard, Nolan and Michael Bowling (2007). “Particle Filtering for Dynamic Agent Modelling in Simplified Poker”. In: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence.* AAAI 2007, pp. 515–521 (cit. on pp. 29, 30, 32, 94).

Bard, Nolan, Michael Johanson, and Michael Bowling (2014). “Asymmetric Abstractions for Adversarial Settings”. In: *Proceedings of the Thirteenth International Conference on Autonomous Agents and Multi-Agent Systems.* AAMAS 2014, pp. 501–508 (cit. on p. 4).

Bard, Nolan, Michael Johanson, Neil Burch, and Michael Bowling (2013). “Online Implicit Agent Modelling”. In: *Proceedings of the Twelfth International Conference on Autonomous Agents and Multi-Agent Systems.* AAMAS 2013, pp. 255–262 (cit. on pp. 3, 67, 69).

Bard, Nolan, Deon Nicholas, Csaba Szepesvári, and Michael Bowling (2015). “Decision-theoretic Clustering of Strategies”. In: *Proceedings of the Fourteenth International Conference on Autonomous Agents and Multi-Agent Systems*. AAMAS 2015, pp. 17–25 (cit. on pp. 4, 59).

Barrett, Samuel and Peter Stone (2015). “Cooperating with Unknown Teammates in Complex Domains: A Robot Soccer Case Study of Ad Hoc Teamwork”. In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI 2015, pp. 2010–2016 (cit. on pp. 32, 35, 39, 93).

Billings, Darse (1995). “Computer Poker”. MSc. Thesis. University of Alberta (cit. on pp. 7, 9).

Billings, Darse (1999). *The First International RoShamBo Programming Competition*. `https://webdocs.cs.ualberta.ca/~darse/rsbpc1.html`. [Online; accessed 02-November-2015] (cit. on p. 75).

Billings, Darse (2006). “Algorithms and Assessment in Computer Poker”. PhD thesis. University of Alberta (cit. on p. 18).

Billings, Darse and Morgan Kan (2006). “A Tool for the Direct Assessment of Poker Decisions”. In: *ICGA Journal* 29.3, pp. 119–142 (cit. on pp. 39, 93).

Billings, Darse, Aaron Davidson, Jonathan Schaeffer, and Duane Szafron (2002). “The challenge of poker”. In: *Artificial Intelligence* 134.1-2, pp. 201–240 (cit. on p. 18).

Billings, Darse, Neil Burch, Aaron Davidson, Robert C. Holte, Jonathan Schaeffer, Terence Schauenberg, and Duane Szafron (2003). “Approximating Game-Theoretic Optimal Strategies for Full-scale Poker”. In: *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*. IJCAI 2003, pp. 661–668 (cit. on pp. 11, 18–20).

Billings, Darse, Aaron Davidson, Terence Schauenberg, Neil Burch, Michael Bowling, Robert Holte, Jonathan Schaeffer, and Duane Szafron (2004). “Game-Tree Search with Adaptation in Stochastic Imperfect-Information Games”. In: *Computers and Games, 4th International Conference, CG 2004, Ramat-Gan, Israel, July 5-7, 2004, Revised Papers*. CG 2004. Vol. 3846, pp. 21–34 (cit. on pp. 29, 30, 32).

Bowling, Michael, Brett Browning, and Manuela Veloso (2004). “Plays as Effective Multiagent Plans Enabling Opponent-Adaptive Play Selection”. In: *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling*. ICAPS 2004, pp. 376–383 (cit. on p. 39).

Bowling, Michael, Michael Johanson, Neil Burch, and Duane Szafron (2008). “Strategy Evaluation in Extensive Games with Importance Sampling”. In: *Proceedings of the Twenty-Fifth International Conference on Machine Learning*. ICML 2008. Vol. 307, pp. 72–79 (cit. on pp. 25, 38, 69, 93).

Bowling, Michael, Neil Burch, Michael Johanson, and Oskari Tammelin (2015). “Heads-up limit holdem poker is solved”. In: *Science* 347.6218, pp. 145–149 (cit. on pp. 13, 16).

Bronowski, Jacob (1973). *The Ascent of Man*. Documentary. Episode 13: The Long Childhood (cit. on p. 7).

Bubeck, Sébastien and Nicolò Cesa-Bianchi (2012). “Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems”. In: *Foundations and Trends in Machine Learning* 5.1, pp. 1–122 (cit. on pp. 21, 94).

Burch, Neil, Michael Johanson, and Michael Bowling (2014). “Solving Imperfect Information Games Using Decomposition”. In: *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. AAAI 2014, pp. 602–608 (cit. on p. 21).

Carlin, Alan and Shlomo Zilberstein (2008). “Value-Based Observation Compression for DEC-POMDPs”. In: *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems.* AAMAS 2008, pp. 501–508 (cit. on p. 53).

Cornuejols, Gerard, Marshall L. Fisher, and George L. Nemhauser (1977). “Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms”. In: *Management science* 23.8, pp. 789–810 (cit. on pp. 27, 57).

Davidson, Aaron (2002). “Opponent Modeling in Poker: Learning and Acting in a Hostile and Uncertain Environment”. MSc. Thesis. University of Alberta (cit. on pp. 32, 33).

Davidson, Joshua (2014). “The Baseline Approach to Agent Evaluation”. MSc. Thesis. University of Alberta (cit. on pp. 24, 93).

Ganzfried, Sam and Tuomas Sandholm (2011). “Game Theory-Based Opponent Modeling in Large Imperfect-Information Games”. In: *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems.* AAMAS 2011, pp. 533–540 (cit. on pp. 29–32, 61, 83).

Ganzfried, Sam and Tuomas Sandholm (2012). “Safe Opponent Exploitation”. In: *Proceedings of the 13th ACM Conference on Electronic Commerce.* EC 2012, pp. 587–604 (cit. on p. 34).

Gibson, Richard (2014). “Regret Minimization in Games and the Development of Champion Multiplayer Computer Poker-Playing Agents”. PhD thesis. University of Alberta (cit. on pp. 80, 81, 126, 127, 131).

Gibson, Richard and Duane Szafron (2011). “On Strategy Stitching in Large Extensive Form Multiplayer Games”. In: *Advances in Neural Information Processing Systems 24: Proceedings of the 25th Annual Conference on Neural Information Processing Systems.* NIPS 2011, pp. 100–108 (cit. on pp. 81, 126).

Gilpin, Andrew and Tuomas Sandholm (2007). “Lossless abstraction of imperfect information games”. In: *Journal of the ACM* 54.5, pp. 295–320 (cit. on p. 11).

Gilpin, Andrew, Tuomas Sandholm, and Troels Bjerre Sørensen (2007). “Potential-aware automated abstraction of sequential games, and holistic equilibrium analysis of Texas Hold’em poker”. In: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence.* AAAI 2007, pp. 50–57 (cit. on pp. 18–20).

Gilpin, Andrew, Tuomas Sandholm, and Troels Bjerre Sørensen (2008). “A heads-up no-limit Texas Hold’em poker player: Discretized betting models and automatically generated equilibrium-finding programs”. In: *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems.* AAMAS 2008, pp. 911–918 (cit. on pp. 20, 21).

Hart, Sergiu and Andreu Mas-Colell (2000). “A Simple Adaptive Procedure Leading to Correlated Equilibrium”. In: *Econometrica* 68.5, pp. 1127–1150 (cit. on p. 12).

Hawkin, John (2014). “Automated Abstraction of Large Action Spaces in Imperfect Information Extensive-Form Games”. PhD thesis. University of Alberta (cit. on p. 20).

Hoda, Samid, Andrew Gilpin, Javier Peña, and Tuomas Sandholm (2010). “Smoothing Techniques for Computing Nash Equilibria of Sequential Games”. In: *Mathematics of Operations Research* 35.2, pp. 494–512 (cit. on p. 11).

Hoehn, Bret (2006). “The Effectiveness of Opponent Modelling in a Small Imperfect Information Game”. MSc. Thesis. University of Alberta (cit. on pp. 30, 39).

Hoehn, Bret, Finnegan Southey, Robert C. Holte, and Valeriy Bulitko (2005). “Effective Short-Term Opponent Exploitation in Simplified Poker”. In: *Proceedings of the Twentieth*

National Conference on Artificial Intelligence. AAAI 2005, pp. 783–788 (cit. on pp. 29, 30, 32).

Johanson, Michael (2007). "Robust Strategies and Counter-Strategies: Building a Champion Level Computer Poker Player". MSc. Thesis. University of Alberta (cit. on pp. 18, 19, 41).

Johanson, Michael (2013). *Measuring the Size of Large No-Limit Poker Games.* Technical Report TR13-01. Department of Computing Science, University of Alberta (cit. on p. 20).

Johanson, Michael and Michael Bowling (2009). "Data Biased Robust Counter Strategies". In: *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics.* AISTATS 2009. Vol. 5, pp. 264–271 (cit. on pp. 15, 29, 30, 34, 42, 46–48, 67, 92).

Johanson, Michael, Martin Zinkevich, and Michael Bowling (2008). "Computing Robust Counter-Strategies". In: *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems.* NIPS 2007, pp. 721–728 (cit. on pp. 3, 14, 15, 18, 19, 29, 30, 32–34, 39, 42, 46, 92, 94).

Johanson, Michael, Kevin Waugh, Michael Bowling, and Martin Zinkevich (2011). "Accelerating Best Response Calculation in Large Extensive Games". In: *Proceedings of the 22nd International Joint Conference on Artificial Intelligence.* IJCAI 2011, pp. 258–265 (cit. on pp. 13, 17, 39, 41, 44, 49, 50, 76, 95).

Johanson, Michael, Nolan Bard, Marc Lanctot, Richard Gibson, and Michael Bowling (2012a). "Efficient Nash Equilibrium Approximation through Monte Carlo Counterfactual Regret Minimization". In: *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems.* AAMAS 2012, pp. 837–846 (cit. on pp. 13, 127).

Johanson, Michael, Nolan Bard, Neil Burch, and Michael Bowling (2012b). "Finding Optimal Abstract Strategies in Extensive-Form Games". In: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence.* AAAI 2012, pp. 1371–1379 (cit. on pp. 42, 44, 52, 92, 95).

Johanson, Michael, Neil Burch, Richard Valenzano, and Michael Bowling (2013). "Evaluating State-Space Abstractions in Extensive-Form Games". In: *Proceedings of the Twelfth International Conference on Autonomous Agents and Multi-Agent Systems.* AAMAS 2013, pp. 271–278 (cit. on pp. 13, 17–19, 41, 67, 126).

Kleinberg, Jon, Christos Papadimitriou, and Prabhakar Raghavan (1998). "A Microeconomic View of Data Mining". In: *Data Mining and Knowledge Discovery* 2.4, pp. 311–324 (cit. on pp. 53–55, 57).

Kleinberg, Jon, Christos Papadimitriou, and Prabhakar Raghavan (2004). "Segmentation Problems". In: *Journal of the ACM* 51.2, pp. 263–280 (cit. on pp. 54, 57).

Kocsis, Levente and Csaba Szepesvári (2005). "Reduced-Variance Payoff Estimation in Adversarial Bandit Problems". In: *Proceedings of the ECML-2005 Workshop on Reinforcement Learning in Non-Stationary Environments* (cit. on pp. 23, 37, 38, 69, 93).

Koller, Daphne, Nimrod Megiddo, and Bernhard von Stengel (1994). "Fast Algorithms for Finding Randomized Strategies in Game Trees". In: *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing.* STOC 1994, pp. 750–759 (cit. on pp. 10, 61).

Krause, Andreas and Daniel Golovin (2014). "Submodular Function Maximization". In: *Tractability: Practical Approaches to Hard Problems* (cit. on p. 26).

Kuhn, H. W. (1950). “A Simplified Two-Person Poker”. In: *Contributions to the Theory of Games* 1, pp. 97–103 (cit. on p. 8).

Lanctot, Marc, Kevin Waugh, Martin Zinkevich, and Michael Bowling (2009). “Monte Carlo Sampling for Regret Minimization in Extensive Games”. In: *Advances in Neural Information Processing Systems 22: Proceedings of the 23rd Annual Conference on Neural Information Processing Systems.* NIPS 2009, pp. 1078–1086 (cit. on pp. 13, 127).

Lloyd, Stuart P. (1982). “Least Squares Quantization in PCM”. In: *IEEE Transactions on Information Theory* 28.2, pp. 129–137 (cit. on p. 61).

Lovász, L. (1983). “Submodular functions and convexity”. In: *Mathematical Programming The State of the Art*, pp. 235–257 (cit. on p. 26).

Lu, Tyler and Craig Boutilier (2011). “Budgeted Social Choice: From Consensus to Personalized Decision Making”. In: *Proceedings of the 22nd International Joint Conference on Artificial Intelligence.* IJCAI 2011, pp. 280–286 (cit. on pp. 53, 58).

McCracken, Peter and Michael Bowling (2004). “Safe Strategies for Agent Modelling in Games”. In: *AAAI Fall Symposium on Artificial Multi-agent Learning* (cit. on pp. 14, 34).

McMahan, H. Brendan and Matthew Streeter (2009). “Tighter Bounds for Multi-Armed Bandits with Expert Advice”. In: *Proceedings of the 22nd Conference on Learning Theory.* COLT 2009 (cit. on p. 94).

Nemhauser, G.L., L.A. Wolsey, and M.L. Fisher (1978). “An analysis of approximations for maximizing submodular set functions – I”. In: *Mathematical Programming* 14.1, pp. 265–294 (cit. on pp. 26, 27, 53, 57, 68).

Pays, Franois (2014). “An Interior Point Approach to Large Games of Incomplete Information”. In: *Proceedings of the Second AAAI Workshop on Computer Poker and Imperfect Information* (cit. on p. 11).

Rubin, Jonathan and Ian Watson (2011). “On Combining Decisions from Multiple Expert Imitators for Performance”. In: *Proceedings of the 22nd International Joint Conference on Artificial Intelligence.* IJCAI 2011, pp. 344–349 (cit. on p. 39).

Rubin, Jonathan and Ian Watson (2012a). “Case-Based Strategies in Computer Poker”. In: *AI Communications* 25.1, pp. 19–48 (cit. on pp. 30, 31, 95).

Rubin, Jonathan and Ian Watson (2012b). “Opponent Type Adaptation for Case-Based Strategies in Adversarial Games”. In: *Case-Based Reasoning Research and Development - 20th International Conference, ICCBR 2012, Lyon, France, September 3-6, 2012. Proceedings.* ICCBR 2012. Vol. 7466, pp. 357–368 (cit. on pp. 29, 30, 33, 35, 83).

Sandholm, Tuomas (2010). “The State of Solving Large Incomplete-Information Games, and Application to Poker”. In: *AI Magazine* 31.4, pp. 13–32 (cit. on p. 44).

Schauenberg, Terence (2006). “Opponent Modelling and Search in Poker”. MSc. Thesis. University of Alberta (cit. on pp. 29–33, 95).

Schnizlein, David (2009). “State Translation in No-Limit Poker”. MSc. Thesis. University of Alberta (cit. on pp. 20, 21, 83, 127).

Schnizlein, David, Michael Bowling, and Duane Szafron (2009). “Probabilistic State Translation in Extensive Games with Large Action Sets”. In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence.* IJCAI 2009, pp. 278–284 (cit. on pp. 95, 127).

Shi, Jiefu and Michael L. Littman (2000). “Abstraction Methods for Game Theoretic Poker”. In: *Computers and Games, Second International Conference, CG 2000, Hamamatsu,* *Japan, October 26-28, 2000, Revised Papers*. CG 2000. Vol. 2063, pp. 333–345 (cit. on p. 19).

Southey, Finnegan, Bret Hoehn, and Robert C. Holte (2009). "Effective short-term opponent exploitation in simplified poker". In: *Machine Learning* 74.2, pp. 159–189 (cit. on pp. 31, 32).

Southey, Finnegan, Michael Bowling, Bryce Larson, Carmelo Piccione, Neil Burch, Darse Billings, and Chris Rayner (2005). "Bayes' Bluff: Opponent Modelling in Poker". In: *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence*. UAI 2005, pp. 550–558 (cit. on pp. 9, 29–32, 35).

Szafron, Duane, Richard Gibson, and Nathan Sturtevant (2013). "A Parameterized Family of Equilibrium Profiles for Three-Player Kuhn Poker". In: *Proceedings of the Twelfth International Conference on Autonomous Agents and Multi-Agent Systems*. AAMAS 2013, pp. 247–254 (cit. on p. 79).

Tammelin, Oskari, Neil Burch, Michael Johanson, and Michael Bowling (2015). "Solving Heads-up Limit Texas Holdem". In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. IJCAI 2015, pp. 645–652 (cit. on p. 13).

von Neumann, John and Oskar Morgenstern (1944). *Theory of Games and Economic Behavior*. Princeton University Press (cit. on p. 7).

Ward, Joe H., Jr. (1963). "Hierarchical Grouping to Optimize an Objective Function". In: *Journal of the American Statistical Association* 58.301, pp. 236–244 (cit. on p. 58).

Waugh, Kevin, Martin Zinkevich, Michael Johanson, Morgan Kan, David Schnizlein, and Michael Bowling (2009a). "A Practical Use of Imperfect Recall". In: *Proceedings of the Eighth Symposium on Abstraction, Reformulation, and Approximation*. SARA 2009 (cit. on pp. 19, 20, 127).

Waugh, Kevin, David Schnizlein, Michael Bowling, and Duane Szafron (2009b). "Abstraction Pathologies in Extensive Games". In: *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems*. AMAS 2009, pp. 781–788 (cit. on pp. 17, 41, 42, 44).

Waugh, Kevin, Dustin Morrill, J. Andrew Bagnell, and Michael Bowling (2015). "Solving Games with Functional Regret Estimation". In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI 2015, pp. 2138–2145 (cit. on p. 16).

White, Martha (2010). "A General Framework for Reducing Variance in Agent Evaluation". MSc. Thesis. University of Alberta (cit. on p. 93).

Wikipedia (2015a). *Betting in poker — Wikipedia, The Free Encyclopedia*. `http://en.wikipedia.org/w/index.php?title=Betting_in_poker&oldid=648042445`. [Online; accessed 25-February-2015] (cit. on p. 8).

Wikipedia (2015b). *List of poker hands — Wikipedia, The Free Encyclopedia*. `http://en.wikipedia.org/w/index.php?title=List_of_poker_hands&oldid=645233645`. [Online; accessed 25-February-2015] (cit. on p. 8).

Zinkevich, Martin, Michael Bowling, and Neil Burch (2007). "A New Algorithm for Generating Equilibria in Massive Zero-Sum Games". In: *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*. AAAI 2007, pp. 788–794 (cit. on pp. 18, 20).

Zinkevich, Martin, Michael Bowling, Nolan Bard, Morgan Kan, and Darse Billings (2006). "Optimal Unbiased Estimators for Evaluating Agent Performance". In: *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. AAAI 2006, pp. 573–579 (cit. on p. 93).

Zinkevich, Martin, Michael Johanson, Michael Bowling, and Carmelo Piccione (2008). "Regret Minimization in Games with Incomplete Information". In: *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems.* NIPS 2007, pp. 1729–1736 (cit. on pp. 11–13, 18, 19, 41, 127).

Appendix A

Annual Computer Poker Competition Results

This appendix presents full cross tables of results from the ACPC total bankroll events discussed in Section 6.2. Each table shows summary statistics of the row player's performance in matches against the column's opponents. For tables displaying winnings, entries specify the row player's expected utility and 95% confidence interval rounded to the nearest milli big blind. Cells are shaded green or red to indicate a positive or negative expected value, respectively, with darker shading indicating that this distinction is statistically significant. The 2012 ACPC computed 95% confidence intervals on each player's total bankroll to evaluate statistical significance, which are included in the winnings table. From 2013 onwards, the ACPC used bootstrapping to compute statistical significance. Confidence tables for these events show the proportion of (1000) bootstrapped samples that the row player had a greater total bankroll than the column player (with empty cells indicating a value of 0). An overview of the tables is below, with results divided by poker variant, competition year, and whether the table presents winnings or confidence results. We present capped and uncapped results when the ACPC's winnings cap of 750 mbb/g impacted an event's results.

Game	Year	Type	Table	Page
Heads-up limit	2012	Winnings with confidence	Table A.1	Page 105
	2013	Capped winnings	Table A.2	Page 106
		Capped confidence	Table A.3	Page 107
		Uncapped winnings	Table A.4	Page 108
		Uncapped confidence	Table A.5	Page 109
	2014	Capped winnings	Table A.6	Page 110
		Capped confidence	Table A.7	Page 111
		Uncapped winnings	Table A.8	Page 112
		Uncapped confidence	Table A.9	Page 113
3-player limit	2013	Winnings	Table A.10	Page 114
		Confidence	Table A.11	Page 115
	2014	Winnings	Table A.12	Page 116
		Confidence	Table A.13	Page 117
Heads-up no-limit	2013	Capped winnings	Table A.14	Page 118
		Capped confidence	Table A.15	Page 119
		Uncapped winnings	Table A.16	Page 120
		Uncapped confidence	Table A.17	Page 121
	2014	Capped winnings	Table A.18	Page 122
		Capped confidence	Table A.19	Page 123
		Uncapped winnings	Table A.20	Page 124
		Uncapped confidence	Table A.21	Page 125

	slumbot	little.rock	zbot	hyperborean.tbr	patience	neo.poker.lab	entropy	sartre	huhuers	feste.tbr	little.ace	AVG
slumbot		18 ± 3	21 ± 4	32 ± 6	26 ± 5	22 ± 5	22 ± 5	19 ± 5	26 ± 5	47 ± 5	55 ± 5	29 ± 3
little.rock	−18 ± 3		−3 ± 3	8 ± 6	7 ± 6	7 ± 6	10 ± 5	17 ± 5	22 ± 5	45 ± 6	40 ± 6	14 ± 3
zbot	−21 ± 4	3 ± 3		18 ± 6	11 ± 5	11 ± 5	3 ± 6	7 ± 5	18 ± 5	36 ± 6	44 ± 6	13 ± 3
hyperborean.tbr	−32 ± 6	−8 ± 6	−18 ± 6		−3 ± 6	−2 ± 5	1 ± 5	6 ± 5	17 ± 6	56 ± 6	52 ± 5	7 ± 2
patience	−26 ± 5	−7 ± 6	−11 ± 5	3 ± 6		−1 ± 6	7 ± 5	7 ± 5	16 ± 5	38 ± 6	41 ± 6	7 ± 3
neo.poker.lab	−22 ± 5	−7 ± 6	−11 ± 5	2 ± 5	1 ± 6		6 ± 6	7 ± 5	10 ± 5	32 ± 6	30 ± 6	5 ± 3
entropy	−22 ± 5	−10 ± 5	−3 ± 6	−1 ± 5	−7 ± 5	−6 ± 6		6 ± 3	8 ± 5	35 ± 5	41 ± 6	4 ± 3
sartre	−19 ± 5	−17 ± 5	−7 ± 5	−6 ± 5	−7 ± 5	−7 ± 5	−6 ± 3		7 ± 5	34 ± 5	50 ± 5	2 ± 3
huhuers	−26 ± 5	−22 ± 5	−18 ± 5	−17 ± 6	−16 ± 5	−10 ± 5	−8 ± 5	−7 ± 5		20 ± 6	22 ± 6	−8 ± 2
feste.tbr	−47 ± 5	−45 ± 6	−36 ± 6	−56 ± 6	−38 ± 6	−32 ± 6	−35 ± 5	−34 ± 5	−20 ± 6		1 ± 6	−34 ± 4
little.ace	−55 ± 5	−40 ± 6	−44 ± 6	−52 ± 5	−41 ± 6	−30 ± 6	−41 ± 6	−50 ± 5	−22 ± 6	−1 ± 6		−38 ± 3

Table A.1: Uncapped winning rates from the 2012 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown.

	marv	feste_tbr	hyperborean_tbr	zbot	littlerock	propokertools	neo_poker_lab	HITSZ_CS_13	chump12	unamur_tbr	liacc	chump1	slugathorus	AVG
marv		70 ± 9	37 ± 8	−2 ± 6	9 ± 7	17 ± 7	−11 ± 6	231 ± 10	218 ± 11	212 ± 9	523 ± 16	417 ± 10	678 ± 12	200
feste_tbr	−70 ± 9		−39 ± 10	−64 ± 10	−57 ± 9	−58 ± 11	−62 ± 9	141 ± 11	118 ± 11	198 ± 10	750 ± 0	540 ± 16	750 ± 0	179
hyperborean_tbr	−37 ± 8	39 ± 10		−26 ± 9	−11 ± 9	−8 ± 9	−32 ± 10	159 ± 9	139 ± 12	182 ± 12	527 ± 21	548 ± 16	617 ± 20	175
zbot	2 ± 6	64 ± 10	26 ± 9		13 ± 7	9 ± 8	−7 ± 7	169 ± 10	152 ± 10	187 ± 10	385 ± 16	365 ± 11	449 ± 15	151
littlerock	−9 ± 7	57 ± 9	11 ± 9	−13 ± 7		6 ± 8	−21 ± 8	151 ± 8	155 ± 9	169 ± 10	393 ± 19	367 ± 13	469 ± 15	144
propokertools	−17 ± 7	58 ± 11	8 ± 9	−9 ± 8	−6 ± 8		−20 ± 7	160 ± 10	152 ± 10	171 ± 9	402 ± 18	353 ± 12	466 ± 17	143
neo_poker_lab	11 ± 6	62 ± 9	32 ± 10	7 ± 7	21 ± 8	20 ± 7		151 ± 9	158 ± 14	184 ± 10	314 ± 18	352 ± 15	405 ± 19	143
HITSZ_CS_13	−231 ± 10	−141 ± 11	−159 ± 9	−169 ± 10	−151 ± 8	−160 ± 10	−151 ± 9		127 ± 21	213 ± 11	609 ± 15	543 ± 19	750 ± 0	90
chump12	−218 ± 11	−118 ± 11	−139 ± 12	−152 ± 10	−155 ± 9	−152 ± 10	−158 ± 14	−127 ± 21		93 ± 12	68 ± 17	677 ± 14	671 ± 15	24
unamur_tbr	−212 ± 9	−198 ± 10	−182 ± 12	−187 ± 10	−169 ± 10	−171 ± 9	−184 ± 10	−213 ± 11	−93 ± 12		132 ± 17	321 ± 13	437 ± 18	−60
liacc	−523 ± 16	−750 ± 0	−527 ± 21	−385 ± 16	−393 ± 19	−402 ± 18	−314 ± 18	−609 ± 15	−68 ± 17	−132 ± 17		602 ± 17	750 ± 0	−229
chump1	−417 ± 10	−540 ± 16	−548 ± 16	−365 ± 11	−367 ± 13	−353 ± 12	−352 ± 15	−543 ± 19	−677 ± 14	−321 ± 13	−602 ± 17		390 ± 19	−391
slugathorus	−678 ± 12	−750 ± 0	−617 ± 20	−449 ± 15	−469 ± 15	−466 ± 17	−405 ± 19	−750 ± 0	−671 ± 15	−437 ± 18	−750 ± 0	−390 ± 19		−569

Table A.2: Capped winning rates from the 2013 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown. Players' winning rates are limited to a maximum of 750 mbb/g.

	marv	feste_tbr	hyperborean_tbr	zbot	littlerock	neo_poker_lab	propokertools
marv		1	1	1	1	1	1
feste_tbr			0.541	1	1	1	1
hyperborean_tbr		0.459		1	1	1	1
zbot					0.995	1	1
littlerock				0.005		0.867	0.924
neo_poker_lab					0.133		0.541
propokertools					0.076	0.459	
HITSZ_CS_13							
chump12							
unamur_tbr							
liacc							
chump1							
slugathorus							

	HITSZ_CS_13	chump12	unamur_tbr	liacc	chump1	slugathorus
marv	1	1	1	1	1	1
feste_tbr	1	1	1	1	1	1
hyperborean_tbr	1	1	1	1	1	1
zbot	1	1	1	1	1	1
littlerock	1	1	1	1	1	1
neo_poker_lab	1	1	1	1	1	1
propokertools	1	1	1	1	1	1
HITSZ_CS_13		1	1	1	1	1
chump12			1	1	1	1
unamur_tbr				1	1	1
liacc					1	1
chump1						1
slugathorus						

Table A.3: Bootstrapped statistical significance analysis from the 2013 ACPC heads-up limit Texas hold'em total bankroll competition. Table values indicate the frequency the row player's (capped) total bankroll was greater than the column player's in the bootstrap samples.

	feste_tbr	marv	hyperborean_tbr	zbot	littlerock	propokertools	neo_poker_lab	HITSZ_CS_13	chump12	unamur_tbr	liacc	chump1	slugathorus	AVG
feste_tbr		−70 ± 9	−39 ± 10	−64 ± 10	−57 ± 9	−58 ± 11	−62 ± 9	141 ± 11	118 ± 11	198 ± 10	802 ± 21	540 ± 16	1064 ± 30	210
marv	70 ± 9		37 ± 8	−2 ± 6	9 ± 7	17 ± 7	−11 ± 6	231 ± 10	218 ± 11	212 ± 9	523 ± 16	417 ± 10	678 ± 12	200
hyperborean_tbr	39 ± 10	−37 ± 8		−26 ± 9	−11 ± 9	−8 ± 9	−32 ± 10	159 ± 9	139 ± 12	182 ± 12	527 ± 21	548 ± 16	617 ± 20	175
zbot	64 ± 10	2 ± 6	26 ± 9		13 ± 7	9 ± 8	−7 ± 7	169 ± 10	152 ± 10	187 ± 10	385 ± 16	365 ± 11	449 ± 15	151
littlerock	57 ± 9	−9 ± 7	11 ± 9	−13 ± 7		6 ± 8	−21 ± 8	151 ± 8	155 ± 9	169 ± 10	393 ± 19	367 ± 13	469 ± 15	144
propokertools	58 ± 11	−17 ± 7	8 ± 9	−9 ± 8	−6 ± 8		−20 ± 7	160 ± 10	152 ± 10	171 ± 9	402 ± 18	353 ± 12	466 ± 17	143
neo_poker_lab	62 ± 9	11 ± 6	32 ± 10	7 ± 7	21 ± 8	20 ± 7		151 ± 9	158 ± 14	184 ± 10	314 ± 18	352 ± 15	405 ± 19	143
HITSZ_CS_13	−141 ± 11	−231 ± 10	−159 ± 9	−169 ± 10	−151 ± 8	−160 ± 10	−151 ± 9		127 ± 21	213 ± 11	609 ± 15	543 ± 19	1031 ± 35	114
chump12	−118 ± 11	−218 ± 11	−139 ± 12	−152 ± 10	−155 ± 9	−152 ± 10	−158 ± 14	−127 ± 21		93 ± 12	68 ± 17	677 ± 14	671 ± 15	24
unamur_tbr	−198 ± 10	−212 ± 9	−182 ± 12	−187 ± 10	−169 ± 10	−171 ± 9	−184 ± 10	−213 ± 11	−93 ± 12		132 ± 17	321 ± 13	437 ± 18	−60
liacc	−802 ± 21	−523 ± 16	−527 ± 21	−385 ± 16	−393 ± 19	−402 ± 18	−314 ± 18	−609 ± 15	−68 ± 17	−132 ± 17		602 ± 17	1564 ± 23	−166
chump1	−540 ± 16	−417 ± 10	−548 ± 16	−365 ± 11	−367 ± 13	−353 ± 12	−352 ± 15	−543 ± 19	−677 ± 14	−321 ± 13	−602 ± 17		390 ± 19	−391
slugathorus	−1064 ± 30	−678 ± 12	−617 ± 20	−449 ± 15	−469 ± 15	−466 ± 17	−405 ± 19	−1031 ± 35	−671 ± 15	−437 ± 18	−1564 ± 23	−390 ± 19		−687

Table A.4: Uncapped winning rates from the 2013 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown.

	feste_tbr	marv	hyperborean_tbr	zbot	littlerock	neo_poker_lab	propokertools
feste_tbr		0.987	1	1	1	1	1
marv	0.013		1	1	1	1	1
hyperborean_tbr				1	1	1	1
zbot					0.999	1	1
littlerock				0.001		0.851	0.924
neo_poker_lab					0.149		0.562
propokertools					0.076	0.438	
HITSZ_CS_13							
chump12							
unamur_tbr							
liacc							
chump1							
slugathorus							

	HITSZ_CS_13	chump12	unamur_tbr	liacc	chump1	slugathorus
feste_tbr	1	1	1	1	1	1
marv	1	1	1	1	1	1
hyperborean_tbr	1	1	1	1	1	1
zbot	1	1	1	1	1	1
littlerock	1	1	1	1	1	1
neo_poker_lab	1	1	1	1	1	1
propokertools	1	1	1	1	1	1
HITSZ_CS_13		1	1	1	1	1
chump12			1	1	1	1
unamur_tbr				1	1	1
liacc					1	1
chump1						1
slugathorus						

Table A.5: Bootstrapped statistical significance analysis of the uncapped 2013 ACPC heads-up limit Texas hold'em total bankroll results. Table values indicate the frequency the row player's uncapped total bankroll was greater than the column player's in the bootstrap samples.

	Escabeche	SmooCT	Hyperborean	Feste	Cleverpiggy	ProPokerTools	652
Escabeche		33 ± 7	7 ± 13	47 ± 6	−0 ± 6	−1 ± 7	−8 ± 8
SmooCT	−33 ± 7		−29 ± 12	26 ± 6	−28 ± 9	−32 ± 7	−26 ± 11
Hyperborean	−7 ± 13	29 ± 12		34 ± 20	−21 ± 8	−6 ± 15	−27 ± 10
Feste	−47 ± 6	−26 ± 6	−34 ± 20		−41 ± 7	−41 ± 8	−52 ± 23
Cleverpiggy	0 ± 6	28 ± 9	21 ± 8	41 ± 7		0 ± 6	−10 ± 9
ProPokerTools	1 ± 7	32 ± 7	6 ± 15	41 ± 8	−0 ± 6		−25 ± 16
652	8 ± 8	26 ± 11	27 ± 10	52 ± 23	10 ± 9	25 ± 16	
Slugathorus	−118 ± 17	−88 ± 15	−72 ± 14	−61 ± 18	−80 ± 20	−71 ± 26	−82 ± 15
Lucifer	−166 ± 9	−152 ± 9	−148 ± 19	−130 ± 8	−129 ± 9	−138 ± 14	−119 ± 12
TerribleTerrance	−158 ± 12	−144 ± 11	−104 ± 15	−110 ± 19	−119 ± 11	−102 ± 32	−92 ± 13
PokerStar	−188 ± 15	−167 ± 10	−190 ± 12	−165 ± 16	−209 ± 12	−203 ± 18	−202 ± 15
HITSZ_CS_14	−266 ± 9	−334 ± 6	−321 ± 22	−242 ± 9	−308 ± 7	−320 ± 9	−304 ± 12
chump9	−520 ± 14	−461 ± 10	−272 ± 25	−614 ± 17	−291 ± 13	−283 ± 12	−268 ± 20
chump4	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−513 ± 15	−501 ± 16	−408 ± 27

	Slugathorus	Lucifer	TerribleTerrance	PokerStar	HITSZ_CS_14	chump9	chump4	AVG
Escabeche	118 ± 17	166 ± 9	158 ± 12	188 ± 15	266 ± 9	520 ± 14	750 ± 0	173
SmooCT	88 ± 15	152 ± 9	144 ± 11	167 ± 10	334 ± 6	461 ± 10	750 ± 0	152
Hyperborean	72 ± 14	148 ± 19	104 ± 15	190 ± 12	321 ± 22	272 ± 25	750 ± 0	143
Feste	61 ± 18	130 ± 8	110 ± 19	165 ± 16	242 ± 9	614 ± 17	750 ± 0	141
Cleverpiggy	80 ± 20	129 ± 9	119 ± 11	209 ± 12	308 ± 7	291 ± 13	513 ± 15	133
ProPokerTools	71 ± 26	138 ± 14	102 ± 32	203 ± 18	320 ± 9	283 ± 12	501 ± 16	129
652	82 ± 15	119 ± 12	92 ± 13	202 ± 15	304 ± 12	268 ± 20	408 ± 27	125
Slugathorus		29 ± 53	35 ± 29	240 ± 24	369 ± 27	750 ± 0	750 ± 0	123
Lucifer	−29 ± 53		6 ± 13	104 ± 15	252 ± 6	233 ± 9	516 ± 10	8
TerribleTerrance	−35 ± 29	−6 ± 13		103 ± 12	200 ± 10	186 ± 17	428 ± 24	4
PokerStar	−240 ± 24	−104 ± 15	−103 ± 12		−43 ± 12	−17 ± 20	667 ± 22	−89
HITSZ_CS_14	−369 ± 27	−252 ± 6	−200 ± 10	43 ± 12		189 ± 8	73 ± 9	−201
chump9	−750 ± 0	−233 ± 9	−186 ± 17	17 ± 20	−189 ± 8		−166 ± 20	−324
chump4	−750 ± 0	−516 ± 10	−428 ± 24	−667 ± 22	−73 ± 9	166 ± 20		−515

Table A.6: Capped winning rates from the 2014 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown. Players' winning rates are limited to a maximum of 750 mbb/g.

	Escabeche	SmooCT	Hyperborean	Feste	Cleverpiggy	ProPokerTools	652	Slugathorus	Lucifer	TerribleTerrance	PokerStar	HITSZ_CS_14	chump9	chump4
Escabeche		1	1	1	1	1	1	1	1	1	1	1	1	1
SmooCT			0.998	1	1	1	1	1	1	1	1	1	1	1
Hyperborean		0.002		0.894	1	1	1	1	1	1	1	1	1	1
Feste			0.106		0.96	0.999	1	1	1	1	1	1	1	1
Cleverpiggy				0.04		0.995	1	0.996	1	1	1	1	1	1
ProPokerTools				0.001	0.005		0.98	0.827	1	1	1	1	1	1
Slugathorus					0.004	0.173	0.571		1	1	1	1	1	1
652						0.02		0.429	1	1	1	1	1	1
Lucifer										0.625	1	1	1	1
TerribleTerrance									0.375		1	1	1	1
PokerStar												1	1	1
HITSZ_CS_14													1	1
chump9														1
chump4														

Table A.7: Bootstrapped statistical significance analysis from the 2014 ACPC heads-up limit Texas hold'em total bankroll competition. Table values indicate the frequency the row player's (capped) total bankroll was greater than the column player's in the bootstrap samples.

	Escabeche	Feste	Slugathorus	SmooCT	Hyperborean	Cleverpiggy	ProPokerTools
Escabeche		47 ± 6	118 ± 17	33 ± 7	7 ± 13	−0 ± 6	−1 ± 7
Feste	−47 ± 6		61 ± 18	−26 ± 6	−34 ± 20	−41 ± 7	−41 ± 8
Slugathorus	−118 ± 17	−61 ± 18		−88 ± 15	−72 ± 14	−80 ± 20	−71 ± 26
SmooCT	−33 ± 7	26 ± 6	88 ± 15		−29 ± 12	−28 ± 9	−32 ± 7
Hyperborean	−7 ± 13	34 ± 20	72 ± 14	29 ± 12		−21 ± 8	−6 ± 15
Cleverpiggy	0 ± 6	41 ± 7	80 ± 20	28 ± 9	21 ± 8		0 ± 6
ProPokerTools	1 ± 7	41 ± 8	71 ± 26	32 ± 7	6 ± 15	−0 ± 6	
652	8 ± 8	52 ± 23	82 ± 15	26 ± 11	27 ± 10	10 ± 9	25 ± 16
Lucifer	−166 ± 9	−130 ± 8	−29 ± 53	−152 ± 9	−148 ± 19	−129 ± 9	−138 ± 14
TerribleTerrance	−158 ± 12	−110 ± 19	−35 ± 29	−144 ± 11	−104 ± 15	−119 ± 11	−102 ± 32
PokerStar	−188 ± 15	−165 ± 16	−240 ± 24	−167 ± 10	−190 ± 12	−209 ± 12	−203 ± 18
HITSZ_CS_14	−266 ± 9	−242 ± 9	−369 ± 27	−334 ± 6	−321 ± 22	−308 ± 7	−320 ± 9
chump9	−520 ± 14	−614 ± 17	−984 ± 35	−461 ± 10	−272 ± 25	−291 ± 13	−283 ± 12
chump4	−769 ± 13	−1099 ± 16	−1050 ± 33	−815 ± 11	−877 ± 29	−513 ± 15	−501 ± 16

	652	Lucifer	TerribleTerrance	PokerStar	HITSZ_CS_14	chump9	chump4	AVG
Escabeche	−8 ± 8	166 ± 9	158 ± 12	188 ± 15	266 ± 9	520 ± 14	769 ± 13	174
Feste	−52 ± 23	130 ± 8	110 ± 19	165 ± 16	242 ± 9	614 ± 17	1099 ± 16	168
Slugathorus	−82 ± 15	29 ± 53	35 ± 29	240 ± 24	369 ± 27	984 ± 35	1050 ± 33	164
SmooCT	−26 ± 11	152 ± 9	144 ± 11	167 ± 10	334 ± 6	461 ± 10	815 ± 11	157
Hyperborean	−27 ± 10	148 ± 19	104 ± 15	190 ± 12	321 ± 22	272 ± 25	877 ± 29	153
Cleverpiggy	−10 ± 9	129 ± 9	119 ± 11	209 ± 12	308 ± 7	291 ± 13	513 ± 15	133
ProPokerTools	−25 ± 16	138 ± 14	102 ± 32	203 ± 18	320 ± 9	283 ± 12	501 ± 16	129
652		119 ± 12	92 ± 13	202 ± 15	304 ± 12	268 ± 20	408 ± 27	125
Lucifer	−119 ± 12		6 ± 13	104 ± 15	252 ± 6	233 ± 9	516 ± 10	8
TerribleTerrance	−92 ± 13	−6 ± 13		103 ± 12	200 ± 10	186 ± 17	428 ± 24	4
PokerStar	−202 ± 15	−104 ± 15	−103 ± 12		−43 ± 12	−17 ± 20	667 ± 22	−89
HITSZ_CS_14	−304 ± 12	−252 ± 6	−200 ± 10	43 ± 12		189 ± 8	73 ± 9	−201
chump9	−268 ± 20	−233 ± 9	−186 ± 17	17 ± 20	−189 ± 8		−166 ± 20	−342
chump4	−408 ± 27	−516 ± 10	−428 ± 24	−667 ± 22	−73 ± 9	166 ± 20		−581

Table A.8: Uncapped winning rates from the 2014 ACPC heads-up limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown.

	Escabeche	Feste	Slugathorus	SmooCT	Hyperborean	Cleverpiggy	ProPokerTools
Escabeche		0.996	0.933	1	1	1	1
Feste	0.004		0.575	1	1	1	1
Slugathorus	0.067	0.425		0.999	1	1	1
SmooCT			0.001		0.823	1	1
Hyperborean				0.177		1	1
Cleverpiggy							0.995
ProPokerTools						0.005	
652							0.015
Lucifer							
TerribleTerrance							
PokerStar							
HITSZ_CS_14							
chump9							
chump4							

	652	Lucifer	TerribleTerrance	PokerStar	HITSZ_CS_14	chump9	chump4
Escabeche	1	1	1	1	1	1	1
Feste	1	1	1	1	1	1	1
Slugathorus	1	1	1	1	1	1	1
SmooCT	1	1	1	1	1	1	1
Hyperborean	1	1	1	1	1	1	1
Cleverpiggy	1	1	1	1	1	1	1
ProPokerTools	0.985	1	1	1	1	1	1
652		1	1	1	1	1	1
Lucifer			0.639	1	1	1	1
TerribleTerrance		0.361		1	1	1	1
PokerStar					1	1	1
HITSZ_CS_14						1	1
chump9							1
chump4							

Table A.9: Bootstrapped statistical significance analysis of the uncapped 2014 ACPC heads-up limit Texas hold'em total bankroll results. Table values indicate the frequency the row player's uncapped total bankroll was greater than the column player's in the bootstrap samples.

	hyperborean_tbr				
	littlerock	neo_poker_lab	HITSZ_CS_13	kempfer	liacc
hyperborean_tbr					
littlerock		−4 ± 9	142 ± 36	94 ± 25	160 ± 48
neo_poker_lab	−45 ± 16		187 ± 34	63 ± 24	−13 ± 54
HITSZ_CS_13	−425 ± 44	−465 ± 37		−202 ± 25	108 ± 59
kempfer	−251 ± 23	−239 ± 28	−182 ± 29		−246 ± 54
liacc	−498 ± 83	−379 ± 82	−560 ± 66	−208 ± 74	

	littlerock			
	neo_poker_lab	HITSZ_CS_13	kempfer	liacc
hyperborean_tbr	49 ± 19	282 ± 21	157 ± 26	338 ± 43
littlerock				
neo_poker_lab		242 ± 19	112 ± 20	1 ± 58
HITSZ_CS_13	−399 ± 41		−118 ± 45	197 ± 55
kempfer	−214 ± 13	−110 ± 36		−188 ± 32
liacc	−248 ± 82	−414 ± 78	−76 ± 63	

	neo_poker_lab			HITSZ_CS_13		kempfer	
hyperborean_tbr	278 ± 27	176 ± 31	391 ± 41	385 ± 35	452 ± 47	455 ± 35	296
littlerock	157 ± 32	103 ± 18	247 ± 44	228 ± 29	216 ± 52	264 ± 46	161
neo_poker_lab				316 ± 23	211 ± 56	101 ± 70	117
HITSZ_CS_13		−216 ± 32	192 ± 46			381 ± 57	−95
kempfer	−100 ± 28		−156 ± 42		−88 ± 60		−177
liacc	−403 ± 68	55 ± 91		−293 ± 59			−302

Table A.10: Capped winning rates from the 2013 ACPC 3-player limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown. Players' winning rates are limited to a maximum of 750 mbb/g.

	hyperborean_tbr	littlerock	neo_poker_lab	HITSZ_CS_13	kempfer	liacc
hyperborean_tbr		1	1	1	1	1
littlerock			1	1	1	1
neo_poker_lab				1	1	1
HITSZ_CS_13					1	1
kempfer						1
liacc						

Table A.11: Bootstrapped statistical significance analysis from the 2013 ACPC 3-player limit Texas hold'em total bankroll competition. Table values indicate the frequency the row player's (capped) total bankroll was greater than the column player's in the bootstrap samples.

	Hyperborean_tbr			
	SmooCT	KEmpfer	HITSZ_CS_14	Lucifer
Hyperborean_tbr				
SmooCT		6 ± 10	66 ± 11	142 ± 16
KEmpfer	−82 ± 10		13 ± 9	147 ± 15
HITSZ_CS_14	−218 ± 12	−189 ± 10		−129 ± 11
Lucifer	−356 ± 17	−388 ± 19	−176 ± 14	

	SmooCT			KEmpfer		HITSZ_CS_14	
	KEmpfer	HITSZ_CS_14	Lucifer	HITSZ_CS_14	Lucifer	Lucifer	AVG
Hyperborean_tbr	76 ± 11	151 ± 8	214 ± 11	176 ± 9	240 ± 14	305 ± 10	194
SmooCT				127 ± 12	219 ± 13	268 ± 14	138
KEmpfer		44 ± 7	170 ± 12			262 ± 13	92
HITSZ_CS_14	−171 ± 11		−120 ± 9		−74 ± 8		−150
Lucifer	−389 ± 16	−149 ± 14		−188 ± 12			−274

Table A.12: Capped winning rates from the 2014 ACPC 3-player limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown. Players' winning rates are limited to a maximum of 750 mbb/g.

	Hyperborean_tbr	SmooCT	KEmpfer	HITSZ_CS_14	Lucifer
Hyperborean_tbr		1	1	1	1
SmooCT			1	1	1
KEmpfer				1	1
HITSZ_CS_14					1
Lucifer					

Table A.13: Bootstrapped statistical significance analysis from the 2014 ACPC 3-player limit Texas hold'em total bankroll competition. Table values indicate the frequency the row player's (capped) total bankroll was greater than the column player's in the bootstrap samples.

	slumbot	hyperborean_tbr	tartanian6	nyx	koypetition	neo_poker_lab	littlerock
slumbot		179 ± 95	190 ± 51	16 ± 31	73 ± 78	121 ± 71	156 ± 41
hyperborean_tbr	−179 ± 95		−105 ± 91	−184 ± 81	−123 ± 90	62 ± 99	−111 ± 84
tartanian6	−190 ± 51	105 ± 91		85 ± 69	122 ± 67	103 ± 74	86 ± 54
nyx	−16 ± 31	184 ± 81	−85 ± 69		64 ± 88	104 ± 62	169 ± 41
koypetition	−73 ± 78	123 ± 90	−122 ± 67	−64 ± 88		279 ± 77	13 ± 81
neo_poker_lab	−121 ± 71	−62 ± 99	−103 ± 74	−104 ± 62	−279 ± 77		−67 ± 68
littlerock	−156 ± 41	111 ± 84	−86 ± 54	−169 ± 41	−13 ± 81	67 ± 68	
entropy	−192 ± 48	−750 ± 0	−175 ± 89	−129 ± 62	−307 ± 100	−206 ± 76	−78 ± 55
Sartre	−311 ± 35	−750 ± 0	−340 ± 58	−179 ± 48	−73 ± 85	−145 ± 48	−160 ± 46
hugh	−378 ± 48	−452 ± 57	−309 ± 88	−370 ± 45	−558 ± 115	−118 ± 121	−427 ± 77
kempfer	−726 ± 59	−750 ± 0	−734 ± 106	−541 ± 62	−750 ± 0	−750 ± 0	−384 ± 51
liacc	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0
HITSZ_CS_13	−656 ± 39	−750 ± 0	−750 ± 0	−574 ± 37	−253 ± 32	−750 ± 0	−251 ± 55

	entropy	Sartre	hugh	kempfer	liacc	HITSZ_CS_13	AVG
slumbot	192 ± 48	311 ± 35	378 ± 48	726 ± 59	750 ± 0	656 ± 39	312
hyperborean_tbr	750 ± 0	750 ± 0	452 ± 57	750 ± 0	750 ± 0	750 ± 0	297
tartanian6	175 ± 89	340 ± 58	309 ± 88	734 ± 106	750 ± 0	750 ± 0	281
nyx	129 ± 62	179 ± 48	370 ± 45	541 ± 62	750 ± 0	574 ± 37	247
koypetition	307 ± 100	73 ± 85	558 ± 115	750 ± 0	750 ± 0	253 ± 32	237
neo_poker_lab	206 ± 76	145 ± 48	118 ± 121	750 ± 0	750 ± 0	750 ± 0	165
littlerock	78 ± 55	160 ± 46	427 ± 77	384 ± 51	750 ± 0	251 ± 55	150
entropy		−65 ± 58	193 ± 69	750 ± 0	750 ± 0	750 ± 0	45
Sartre	65 ± 58		143 ± 60	750 ± 0	750 ± 0	750 ± 0	42
hugh	−193 ± 69	−143 ± 60		750 ± 0	750 ± 0	750 ± 0	−58
kempfer	−750 ± 0	−750 ± 0	−750 ± 0		750 ± 0	750 ± 0	−449
liacc	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0		750 ± 0	−625
HITSZ_CS_13	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0		−645

Table A.14: Capped winning rates from the 2013 ACPC heads-up no-limit Texas hold'em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown. Players' winning rates are limited to a maximum of 750 mbb/g.

	slumbot	hyperborean_tbr	tartanian6	nyx	koypetition	neo_poker_lab	littlerock
slumbot		0.85	0.969	1	1	1	1
hyperborean_tbr	0.15		0.65	1	0.999	1	1
tartanian6	0.031	0.35		1	0.997	1	1
nyx					0.6	1	1
koypetition		0.001	0.003	0.4		1	1
neo_poker_lab							0.831
littlerock						0.169	
entropy							
Sartre							
hugh							
kempfer							
liacc							
HITSZ_CS_13							

	entropy	Sartre	hugh	kempfer	liacc	HITSZ_CS_13
slumbot	1	1	1	1	1	1
hyperborean_tbr	1	1	1	1	1	1
tartanian6	1	1	1	1	1	1
nyx	1	1	1	1	1	1
koypetition	1	1	1	1	1	1
neo_poker_lab	1	1	1	1	1	1
littlerock	1	1	1	1	1	1
entropy		0.825	1	1	1	1
Sartre	0.175		1	1	1	1
hugh				1	1	1
kempfer					1	1
liacc						1
HITSZ_CS_13						

Table A.15: Bootstrapped statistical significance analysis from the 2013 ACPC heads-up no-limit Texas hold'em total bankroll competition. Table values indicate the frequency the row player's (capped) total bankroll was greater than the column player's in the bootstrap samples.

	hyperborean_tbr	neo_poker_lab	koypetition	tartanian6	slumbot	nyx	entropy
hyperborean_tbr		62 ± 99	−123 ± 90	−105 ± 91	−179 ± 95	−184 ± 81	1157 ± 132
neo_poker_lab	−62 ± 99		−279 ± 77	−103 ± 74	−121 ± 71	−104 ± 62	206 ± 76
koypetition	123 ± 90	279 ± 77		−122 ± 67	−73 ± 78	−64 ± 88	307 ± 100
tartanian6	105 ± 91	103 ± 74	122 ± 67		−190 ± 51	85 ± 69	175 ± 89
slumbot	179 ± 95	121 ± 71	73 ± 78	190 ± 51		16 ± 31	192 ± 48
nyx	184 ± 81	104 ± 62	64 ± 88	−85 ± 69	−16 ± 31		129 ± 62
entropy	−1157 ± 132	−206 ± 76	−307 ± 100	−175 ± 89	−192 ± 48	−129 ± 62	
Sartre	−974 ± 179	−145 ± 48	−73 ± 85	−340 ± 58	−311 ± 35	−179 ± 48	65 ± 58
hugh	−452 ± 57	−118 ± 121	−558 ± 115	−309 ± 88	−378 ± 48	−370 ± 45	−193 ± 69
littlerock	111 ± 84	67 ± 68	−13 ± 81	−86 ± 54	−156 ± 41	−169 ± 41	78 ± 55
kempfer	−1112 ± 165	−1073 ± 88	−1186 ± 118	−734 ± 106	−726 ± 59	−541 ± 62	−1166 ± 88
liacc	−5709 ± 384	−6111 ± 282	−3803 ± 299	−2815 ± 196	−2701 ± 161	−2735 ± 160	−2661 ± 196
HITSZ_CS_13	−8927 ± 565	−2081 ± 40	−253 ± 32	−1030 ± 66	−656 ± 39	−574 ± 37	−2180 ± 74

	Sartre	hugh	littlerock	kempfer	liacc	HITSZ_CS_13	AVG
hyperborean_tbr	974 ± 179	452 ± 57	−111 ± 84	1112 ± 165	5709 ± 384	8927 ± 565	1474
neo_poker_lab	145 ± 48	118 ± 121	−67 ± 68	1073 ± 88	6111 ± 282	2081 ± 40	750
koypetition	73 ± 85	558 ± 115	13 ± 81	1186 ± 118	3803 ± 299	253 ± 32	528
tartanian6	340 ± 58	309 ± 88	86 ± 54	734 ± 106	2815 ± 196	1030 ± 66	476
slumbot	311 ± 35	378 ± 48	156 ± 41	726 ± 59	2701 ± 161	656 ± 39	475
nyx	179 ± 48	370 ± 45	169 ± 41	541 ± 62	2735 ± 160	574 ± 37	412
entropy	−65 ± 58	193 ± 69	−78 ± 55	1166 ± 88	2661 ± 196	2180 ± 74	324
Sartre		143 ± 60	−160 ± 46	893 ± 73	2387 ± 228	2467 ± 512	315
hugh	−143 ± 60		−427 ± 77	1952 ± 111	3033 ± 339	1075 ± 34	259
littlerock	160 ± 46	427 ± 77		384 ± 51	1689 ± 163	251 ± 55	229
kempfer	−893 ± 73	−1952 ± 111	−384 ± 51		1767 ± 277	2547 ± 55	−454
liacc	−2387 ± 228	−3033 ± 339	−1689 ± 163	−1767 ± 277		19784 ± 260	−1302
HITSZ_CS_13	−2467 ± 512	−1075 ± 34	−251 ± 55	−2547 ± 55	−19784 ± 260		−3486

Table A.16: Uncapped winning rates from the 2013 ACPC heads-up no-limit Texas hold’em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown.

	hyperborean_tbr	neo_poker_lab	koypetition	tartanian6	slumbot	nyx	entropy	Sartre	hugh	littlerock	kempfer	liacc	HITSZ_CS_13
hyperborean_tbr		1	1	1	1	1	1	1	1	1	1	1	1
neo_poker_lab			1	1	1	1	1	1	1	1	1	1	1
koypetition				0.891	0.984	1	1	1	1	1	1	1	1
tartanian6			0.109		0.79	1	1	1	1	1	1	1	1
slumbot			0.016	0.21		1	1	1	1	1	1	1	1
nyx							1	1	1	1	1	1	1
entropy								0.958	1	1	1	1	1
Sartre							0.042		0.833	0.94	1	1	1
hugh								0.167		0.763	1	1	1
littlerock								0.06	0.237		1	1	1
kempfer												1	1
liacc													1
HITSZ_CS_13													

Table A.17: Bootstrapped statistical significance analysis of the uncapped 2013 ACPC heads-up no-limit Texas hold'em total bankroll results. Table values indicate the frequency the row player's uncapped total bankroll was greater than the column player's in the bootstrap samples.

	Tartanian7	Nyx	Prelude	Slumbot	HibiscusBiscuit	Feste_tbr	Hyperborean_tbr
Tartanian7		121 ± 38	20 ± 16	33 ± 16	125 ± 44	201 ± 44	212 ± 113
Nyx	−121 ± 38		−59 ± 52	−64 ± 43	21 ± 52	56 ± 41	194 ± 136
Prelude	−20 ± 16	59 ± 52		−6 ± 15	109 ± 60	174 ± 72	229 ± 117
Slumbot	−33 ± 16	64 ± 43	6 ± 15		94 ± 48	143 ± 67	199 ± 75
HibiscusBiscuit	−125 ± 44	−21 ± 52	−109 ± 60	−94 ± 48		33 ± 48	112 ± 100
Feste_tbr	−201 ± 44	−56 ± 41	−174 ± 72	−143 ± 67	−33 ± 48		77 ± 110
Hyperborean_tbr	−212 ± 113	−194 ± 136	−229 ± 117	−199 ± 75	−112 ± 100	−77 ± 110	
LittleRock	−214 ± 57	−310 ± 70	−272 ± 64	−109 ± 49	−139 ± 72	−22 ± 51	−61 ± 67
SartreNLExp	−261 ± 47	−146 ± 59	−116 ± 45	−199 ± 31	−42 ± 69	−58 ± 46	117 ± 106
PijaiBot	−499 ± 68	−687 ± 68	−205 ± 64	−298 ± 73	−427 ± 87	−275 ± 101	−181 ± 93
Rembrant3	−750 ± 0	−596 ± 49	−750 ± 0	−750 ± 0	−685 ± 56	−750 ± 0	−750 ± 0
KEmpfer	−516 ± 61	−750 ± 0	−597 ± 91	−627 ± 51	−540 ± 71	−493 ± 76	−750 ± 0
HITSZ_CS_14	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0
Lucifer	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0

	LittleRock	SartreNLExp	PijaiBot	Rembrant3	KEmpfer	HITSZ_CS_14	Lucifer	AVG
Tartanian7	214 ± 57	261 ± 47	499 ± 68	750 ± 0	516 ± 61	750 ± 0	750 ± 0	342
Nyx	310 ± 70	146 ± 59	687 ± 68	596 ± 49	750 ± 37	750 ± 0	750 ± 0	309
Prelude	272 ± 64	116 ± 45	205 ± 64	750 ± 0	597 ± 91	750 ± 0	750 ± 0	306
Slumbot	109 ± 49	199 ± 31	298 ± 73	750 ± 0	627 ± 51	750 ± 0	750 ± 0	304
HibiscusBiscuit	139 ± 72	42 ± 69	427 ± 87	685 ± 56	540 ± 71	750 ± 0	750 ± 0	241
Feste_tbr	22 ± 51	58 ± 46	275 ± 101	750 ± 0	493 ± 76	750 ± 0	750 ± 0	198
Hyperborean_tbr	61 ± 67	−117 ± 106	181 ± 93	750 ± 0	750 ± 0	750 ± 0	750 ± 0	162
LittleRock		60 ± 45	238 ± 78	750 ± 0	506 ± 110	676 ± 253	750 ± 0	143
SartreNLExp	−60 ± 45		103 ± 115	102 ± 41	750 ± 0	750 ± 0	750 ± 0	130
PijaiBot	−238 ± 78	−103 ± 115		750 ± 0	750 ± 0	750 ± 0	750 ± 0	7
Rembrant3	−750 ± 0	−102 ± 41	−750 ± 0		−374 ± 86	750 ± 0	750 ± 0	−424
KEmpfer	−506 ± 110	−750 ± 0	−750 ± 0	374 ± 86		−750 ± 0	750 ± 0	−454
HITSZ_CS_14	−676 ± 253	−750 ± 0	−750 ± 0	−750 ± 0	750 ± 0		−750 ± 0	−629
Lucifer	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	−750 ± 0	750 ± 0		−635

Table A.18: Capped winning rates from the 2014 ACPC heads-up no-limit Texas hold’em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown. Players’ winning rates are limited to a maximum of 750 mbb/g.

	Tartanian7	Nyx	Prelude	Slumbot	HibiscusBiscuit	Feste_tbr	Hyperborean_tbr
Tartanian7		0.999	1	1	1	1	1
Nyx	0.001		0.577	0.804	1	1	1
Prelude		0.423		0.835	1	1	1
Slumbot		0.196	0.165		1	1	1
HibiscusBiscuit						1	1
Feste_tbr							0.994
Hyperborean_tbr						0.006	
LittleRock							0.059
SartreNLExp							0.002
PijaiBot							
Rembrant3							
KEmpfer							
HITSZ_CS_14							
Lucifer							

	LittleRock	SartreNLExp	PijaiBot	Rembrant3	KEmpfer	HITSZ_CS_14	Lucifer
Tartanian7	1	1	1	1	1	1	1
Nyx	1	1	1	1	1	1	1
Prelude	1	1	1	1	1	1	1
Slumbot	1	1	1	1	1	1	1
HibiscusBiscuit	1	1	1	1	1	1	1
Feste_tbr	1	1	1	1	1	1	1
Hyperborean_tbr	0.941	0.998	1	1	1	1	1
LittleRock		0.945	1	1	1	1	1
SartreNLExp	0.055		1	1	1	1	1
PijaiBot				1	1	1	1
Rembrant3					1	1	1
KEmpfer						1	1
HITSZ_CS_14							1
Lucifer							

Table A.19: Bootstrapped statistical significance analysis from the 2014 ACPC heads-up no-limit Texas hold'em total bankroll competition. Table values indicate the frequency the row player's (capped) total bankroll was greater than the column player's in the bootstrap samples.

	Hyperborean_tbr	SartreNLExp	Nyx	Tartanian7	Slumbot	Prelude	HibiscusBiscuit	Feste_tbr	PijaiBot	LittleRock	KEmpfer	Rembrant3	HITSZ_CS_14	Lucifer	AVG
Hyperborean_tbr		−117 ± 106	−194 ± 136	−212 ± 113	−199 ± 75	−229 ± 117	−112 ± 100	−77 ± 110	181 ± 93	61 ± 67	2142 ± 257	1268 ± 123	4883 ± 986	4877 ± 696	944
SartreNLExp	117 ± 106		−146 ± 59	−261 ± 47	−199 ± 31	−116 ± 45	−42 ± 69	−58 ± 46	103 ± 115	−60 ± 45	1235 ± 67	102 ± 41	4250 ± 471	4085 ± 117	693
Nyx	194 ± 136	146 ± 59		−121 ± 38	−64 ± 43	−59 ± 52	21 ± 52	56 ± 41	687 ± 68	310 ± 70	770 ± 57	596 ± 49	2156 ± 156	3063 ± 111	597
Tartanian7	212 ± 113	261 ± 47	121 ± 38		33 ± 16	20 ± 16	125 ± 44	201 ± 44	499 ± 68	214 ± 57	516 ± 61	980 ± 34	1474 ± 180	1819 ± 111	498
Slumbot	199 ± 75	199 ± 31	64 ± 43	−33 ± 16		6 ± 15	94 ± 48	143 ± 67	298 ± 73	109 ± 49	627 ± 51	868 ± 50	1570 ± 139	1967 ± 72	470
Prelude	229 ± 117	116 ± 45	59 ± 52	−20 ± 16	−6 ± 15		109 ± 60	174 ± 72	205 ± 64	272 ± 64	597 ± 91	845 ± 44	1361 ± 238	2164 ± 119	470
HibiscusBiscuit	112 ± 100	42 ± 69	−21 ± 52	−125 ± 44	−94 ± 48	−109 ± 60		33 ± 48	427 ± 87	139 ± 72	540 ± 71	685 ± 56	1127 ± 206	1988 ± 109	365
Feste_tbr	77 ± 110	58 ± 46	−56 ± 41	−201 ± 44	−143 ± 67	−174 ± 72	−33 ± 48		275 ± 101	22 ± 51	493 ± 76	875 ± 55	1089 ± 198	2382 ± 96	359
PijaiBot	−181 ± 93	−103 ± 115	−687 ± 68	−499 ± 68	−298 ± 73	−205 ± 64	−427 ± 87	−275 ± 101		−238 ± 78	1026 ± 89	1400 ± 123	1785 ± 290	2942 ± 119	326
LittleRock	−61 ± 67	60 ± 45	−310 ± 70	−214 ± 57	−109 ± 49	−272 ± 64	−139 ± 72	−22 ± 51	238 ± 78		506 ± 110	853 ± 49	676 ± 253	1415 ± 80	202
KEmpfer	−2142 ± 257	−1235 ± 67	−770 ± 57	−516 ± 61	−627 ± 51	−597 ± 91	−540 ± 71	−493 ± 76	−1026 ± 89	−506 ± 110		374 ± 86	−1018 ± 261	5827 ± 166	−252
Rembrant3	−1268 ± 123	−102 ± 41	−596 ± 49	−980 ± 34	−868 ± 50	−845 ± 44	−685 ± 56	−875 ± 55	−1400 ± 123	−853 ± 49	−374 ± 86		1646 ± 130	1790 ± 101	−416
HITSZ_CS_14	−4883 ± 986	−4250 ± 471	−2156 ± 156	−1474 ± 180	−1570 ± 139	−1361 ± 238	−1127 ± 206	−1089 ± 198	−1785 ± 290	−676 ± 253	1018 ± 261	−1646 ± 130		−2412 ± 515	−1801
Lucifer	−4877 ± 696	−4085 ± 117	−3063 ± 111	−1819 ± 111	−1967 ± 72	−2164 ± 119	−1988 ± 109	−2382 ± 96	−2942 ± 119	−1415 ± 80	−5827 ± 166	−1790 ± 101	2412 ± 515		−2455

Table A.20: Uncapped winning rates from the 2014 ACPC heads-up no-limit Texas hold’em total bankroll competition. Results are in mbb/g with 95% confidence intervals shown.

	Hyperborean_tbr	SartreNLExp	Nyx	Tartanian7	Prelude	Slumbot	HibiscusBiscuit
Hyperborean_tbr		1	1	1	1	1	1
SartreNLExp			1	1	1	1	1
Nyx				1	1	1	1
Tartanian7					0.998	1	1
Prelude				0.002		0.525	1
Slumbot					0.475		1
HibiscusBiscuit							
Feste_tbr							0.35
PijaiBot							0.098
LittleRock							
KEmpfer							
Rembrant3							
HITSZ_CS_14							
Lucifer							

	Feste_tbr	PijaiBot	LittleRock	KEmpfer	Rembrant3	HITSZ_CS_14	Lucifer
Hyperborean_tbr	1	1	1	1	1	1	1
SartreNLExp	1	1	1	1	1	1	1
Nyx	1	1	1	1	1	1	1
Tartanian7	1	1	1	1	1	1	1
Prelude	1	1	1	1	1	1	1
Slumbot	1	1	1	1	1	1	1
HibiscusBiscuit	0.65	0.902	1	1	1	1	1
Feste_tbr		0.895	1	1	1	1	1
PijaiBot	0.105		1	1	1	1	1
LittleRock				1	1	1	1
KEmpfer					1	1	1
Rembrant3						1	1
HITSZ_CS_14							1
Lucifer							

Table A.21: Bootstrapped statistical significance analysis of the uncapped 2014 ACPC heads-up no-limit Texas hold'em total bankroll results. Table values indicate the frequency the row player's uncapped total bankroll was greater than the column player's in the bootstrap samples.

Appendix B

Annual Computer Poker Competition Agents

This appendix provides implementation details of the Hyperborean agents discussed in Section 6.2 for our analysis of prior Annual Computer Poker Competition results. Agents are divided by poker variant, competition year, and event. Our implicit modelling agents were submitted to the total bankroll event, while agents submitted to the bankroll instant runoff events were prepared by other members of the University of Alberta's Computer Poker Research Group. We detail each agent with a brief description and a listing of any strategies used by the agent. For each strategy, we provide a high-level description and summarize notable implementation details in several categories. Additional details specifying how we use an implicit modelling agent's portfolio online are presented in Section 6.1.1. We describe the categories of implementation details next, however, note that some may not apply to every strategy.

Abstraction. This category describes the abstract games used to generate the strategy. A **symmetric** abstraction represents every player's view of the game using a single abstract game. In contrast, **asymmetric** abstractions (introduced in Chapter 4) use multiple abstract games to represent the agent's strategy, an opponent's regret minimizing response, or an opponent's frequentist model in a DBR. In the case of **dynamic expert** strategies (Gibson and Szafron 2011; Gibson 2014, Section 7.2), an abstract game is defined by specifying a partitioning of the game along with the different abstractions used for each part. Each abstract game is defined in terms of the abstractions used for the **cards** and **betting**, whose notation we define next.

Card abstraction (Section 2.5.1) is used for each strategy. Most of the card abstractions used by these strategies are similar to Johanson and colleagues' (2013) "IR KE-KO 169-9000-9000-9000" abstraction. This imperfect recall (IR) abstraction clusters cards into 9000 buckets using k-means for each round after the preflop, entirely forgetting earlier card information. On the flop and turn, clusters are formed based on the earth mover's distance between hand strength distributions (KE), while opponent cluster hand strength features (KO) are used for clustering on the river. On the preflop, 169 buckets are used to exactly distinguish each canonical pair of private cards. We also use abstract games based on other card abstraction techniques. For instance, "IR KE-PHS 169-900-100-25" substitutes percentile hand strength (PHS) buckets for the KO buckets on the river. Finally, we also use abstract games that independently characterize the texture of the public cards using public buckets. For example, "IR PUB×KE-KO 169-1348620-51×30000-280×3000" specifies an imperfect recall abstraction with all the canonical cards on the preflop and flop, 51 public buckets combined with 30000 KE buckets on the turn, and 280 public buckets combined with 3000 KO buckets on the river. Waugh and colleagues (2009a) provide one approach to producing public buckets, however the specific technique used in these abstractions has not been published and is outside the scope of this dissertation.

Betting abstraction (Section 2.5.2) is used exclusively in heads-up no-limit. For all poker variants, the abstract games used have perfect recall of past (potentially abstract) betting. In heads-up no-limit, we restrict raising actions to a set of magnitudes relative to the pot size. Aside from these pot fraction raises, the smallest and largest legal raises, a “min raise” and “all-in” respectively, may be available. Although we list the bet sizes used in the abstraction, there may be restrictions on when and how frequently each bet size may be used. We omit some details regarding these restrictions for simplicity.

CFR technique. This category describes the CFR algorithm used to generate the strategy. We specify which of the following CFR variants was used to generate each strategy: chance sampled (Zinkevich et al. 2008), public chance sampled (Johanson et al. 2012a), external sampled (Lanctot et al. 2009), or Pure CFR (Gibson 2014, Section 5.5). Furthermore, we list the approximate number of **iterations** the CFR variant was run, and whether the average or current **strategy profile** was used. Note that although the current strategy profile typically lacks convergence guarantees, Gibson (2014, Section 4.4.3) demonstrated that its one-on-one performance can improve much more quickly than the average strategy profile.

Miscellaneous categories. Finally, there are several categories that do not appear for every strategy. For data biased responses, we list the $P_{\max}$ parameter value that the DBR uses with the 0-10 linear function (see Section 2.4.3). For heads-up no-limit strategies, we specify the **translation** scheme used to map real-game betting sequences into the abstract game. We use either hard or soft (*i.e.*, deterministic or probabilistic) translation with a geometric similarity function (Schnizlein, Bowling, and Szafron 2009; Schnizlein 2009). Lastly, we list any uncategorized **strategy notes**.

B.1 Heads-up Limit Texas Hold'em

B.1.1 2012

Total Bankroll

The implicit modelling agent submitted to the 2012 total bankroll competition used a portfolio consisting of seven abstract strategies. The responses to always raise and always call were only used when the opponent was detected to be always raise or always call. Otherwise, the agent employed an implicit modelling approach: combining a portfolio of the remaining five strategies.

1–2. Counter-strategies to opponents that always raise or always call

  Abstraction: symmetric

    Cards: IR KE-KO 169-9000-9000-9000

  CFR technique: chance sampled

    Iterations: 1 billion

    Strategy profile: average

3. An abstract Nash equilibrium approximation

  Abstraction: symmetric

    Cards: IR PUB×KE-KO 169-1348620-60×9000-60×1308, with partial recall of public card texture

  CFR technique: chance sampled

    Iterations: 43 billion

    Strategy profile: average

4–7. Data biased responses to asymmetric models of particular opponents seen in the 2010 (ASVP) or 2011 ACPC (RobotBot, TellBot, Tiltnet)

  DBR $P_{\max}$: 0.2 (RobotBot), 0.4 (TellBot), 0.5 (Tiltnet), 0.6 (ASVP)

  Abstraction: asymmetric

    Cards: IR KE-KO 169-9000-9000-9000

    Frequentist model cards: 5-bucket, perfect recall, percentile $E[HS^2]$ (described in Section 4.1.1).

  CFR technique: chance sampled

    Iterations: 2 billion

    Strategy profile: average

B.1.2 2013

Total Bankroll

The implicit modelling agent submitted to the 2013 total bankroll competition used a portfolio consisting of four abstract strategies.

1–4. Data biased responses to asymmetric models of particular opponents seen in the 2010 (ASVP) or 2011 ACPC (RobotBot, TellBot, Tiltnet)

DBR $P_{\max}$: 0.4 (RobotBot), 0.5 (TellBot), 0.7 (Tiltnet, ASVP)

Abstraction: asymmetric

Cards: IR KE-KO 169-18630-18630-18630

Frequentist model cards: IR KE-KO 169-3700-3700-3700

CFR technique: chance sampled

Iterations: 4 billion

Strategy profile: average

Bankroll Instant Runoff

1. An abstract Nash equilibrium approximation

Abstraction: symmetric

Cards: IR PUB×KE-KO 169-1348620-51×30000-280×3000

CFR technique: chance sampled

Iterations: 130 billion

Strategy profile: average

B.1.3 2014

Total Bankroll

The implicit modelling agent submitted to the 2014 total bankroll competition used a portfolio consisting of four abstract strategies.

1–2. The data biased responses to Tiltnet and ASVP from the 2013 portfolio

DBR P_{max}: 0.7 (Tiltnet, ASVP)

Abstraction: asymmetric

Cards: IR KE-KO 169-18630-18630-18630

Frequentist model cards: IR KE-KO 169-3700-3700-3700

CFR technique: chance sampled

Iterations: 4 billion

Strategy profile: average

3. Data biased response to data from Feste agents since 2011

DBR P_{max}: 0.75

Abstraction: asymmetric

Cards: IR KE-KO 169-1348620-34470-34470

Frequentist model cards: IR KE-KO 169-3700-3700-3700

CFR technique: public chance sampled

Iterations: 10 million

Strategy profile: average

4. An asymmetric abstract Nash equilibrium approximation (designed to exploit mistakes made by equilibrium-based agents using smaller abstractions of the game)

Abstraction: asymmetric

Strategy Cards: IR PUB×KE-KO 169-1348620-51×30000-280×3000

Opponent Cards: IR KE-KO 169-9000-9000-9000

CFR technique: public chance sampled

Iterations: 48 million

Strategy profile: average

B.2 3-player Limit Texas Hold'em

B.2.1 2013 & 2014

Total Bankroll

Our 2013 and 2014 total bankroll agent was not an implicit modelling agent and only used a single strategy rather than a portfolio of strategies.

1. A data biased response to aggregate data of all ACPC competitors from the 2011 and 2012 3-player limit competitions

   DBR $P_{\max}$: 0.5

   Abstraction: asymmetric

   - **Cards:** IR KE-KO 169-10000-5450-500
   - **Frequentist model cards:** IR KE-PHS 169-900-100-25

   CFR technique: external sampled

   - **Iterations:** 20 billion
   - **Strategy profile:** current

   Strategy notes: Both opponents used the same card abstraction and frequentist model when training the DBR in each of the three player positions.

Bankroll Instant Runoff

Richard Gibson (2014, Section 8.4) designed the dynamic expert strategy used for the agent submitted to both the 2013 and 2014 bankroll instant runoff events.

1. Dynamic expert abstract Nash equilibrium approximation

   Abstraction: symmetric, dynamic expert

   - **Dynamic experts:** two granularities of card abstractions for "important" and "unimportant" betting sequences. Sequences partitioned based on the pot size and the frequency that the Hyperborean agents from the 2011 and 2012 ACPCs observed the betting sequence.
   - **Cards:** IR PUB×KE-KO 169-9×20000-18630-875 (unimportant),
     IR PUB×KE-KO 169-1348620-51×30000-280×10000 (important)

   CFR technique: Pure CFR

   - **Iterations:** 303.6 billion
   - **Strategy profile:** current

B.3 Heads-up No-limit Texas Hold'em

B.3.1 2013

Total Bankroll

Our implicit modelling agent for the 2013 ACPC total bankroll event uses a portfolio of two abstract strategies.

1. Dynamic expert abstract Nash equilibrium approximation

   Abstraction: symmetric, dynamic expert

   Dynamic experts: two granularities of card abstractions for "important" and "unimportant" betting sequences. Sequences partitioned based on the pot size and the frequency that an earlier heads-up no-limit agent observed the betting sequence in self-play.

   Cards: IR KE-KO 169-3700-3700-3700 (unimportant),
   IR PUB×KE-KO 169-9×20000-51×30000-280×6000 (important)

   Betting: 0.5, 0.75, 1, 1.5, 3, 6, 11, 20, or 40 times the pot size, or all-in

   CFR technique: Pure CFR

   Iterations: 498 billion

   Strategy profile: current

   Translation: Hard, geometric similarity

2. A data biased response to aggregate data from all of the agents in the 2011 and 2012 heads-up no-limit ACPC events

   DBR P_{max}: 0.25

   Abstraction: asymmetric

   Cards: IR KE-KO 169-9000-9000-3700

   Frequentist model cards: 1 bucket (ignores cards)

   Betting: 0.5, 0.75, 1, 1.5, 3, 6, 11, 20, or 40 times the pot size, or all-in

   CFR technique: external sampled

   Iterations: 84 billion

   Strategy profile: average

   Translation: Hard, geometric similarity

   Strategy notes: Hard translation based on geometric similarity was also used to build the frequentist model from the real game observations.

Bankroll Instant Runoff

This agent is a meta-player that switches between 2 different strategies. A default strategy is played until we have seen the opponent make a minimum-sized bet on at least 1% of the hands played so far (a min bet as the first bet of the game is not counted). At this time, we switch to an alternative strategy that both makes min bets itself and better understands min bets.

1. Dynamic expert abstract Nash equilibrium approximation

    Abstraction: symmetric, dynamic expert

    - **Dynamic experts:** two granularities of card abstractions for "important" and "unimportant" betting sequences. Sequences partitioned based on the pot size and the frequency that an earlier heads-up no-limit agent observed the betting sequence in self-play.
    - **Cards:** IR KE-KO 169-3700-3700-3700 (unimportant),
IR PUB×KE-KO 169-9×20000-51×30000-280×6000 (important)
    - **Betting:** 0.5, 0.75, 1, 1.5, 3, 6, 11, 20, or 40 times the pot size, or all-in

    CFR technique: Pure CFR

    - **Iterations:** 498 billion
    - **Strategy profile:** current

    Translation: Soft, geometric similarity

2. An abstract Nash equilibrium approximation that both makes min bets itself and better understands min bets

    Abstraction: symmetric

    - **Cards:** IR KE-KO 169-3700-3700-1175
    - **Betting:** min, 0.5, 0.75, 1, 2, 3, or 11 times the pot size, or all-in

    CFR technique: external sampled

    - **Iterations:** 132 billion
    - **Strategy profile:** current

    Translation: Soft, geometric similarity

    Strategy notes: Each player is only allowed one 0.5 and one 0.75 bet in a round, and not on the preflop. Minimum sized bets must be the first action in a round.

B.3.2 2014

Total Bankroll

Our implicit modelling agent for the 2014 ACPC total bankroll event uses a portfolio of three abstract strategies, including two from the 2013 total bankroll portfolio.

1. Dynamic expert abstract Nash equilibrium approximation

   Abstraction: symmetric, dynamic expert

   Dynamic experts: two granularities of card abstractions for "important" and "unimportant" betting sequences. Sequences partitioned based on the pot size and the frequency that an earlier heads-up no-limit agent observed the betting sequence in self-play.

   Cards: IR KE-KO 169-3700-3700-3700 (unimportant),
   IR PUB×KE-KO 169-9×20000-51×30000-280×6000 (important)

   Betting: 0.5, 0.75, 1, 1.5, 3, 6, 11, 20, or 40 times the pot size, or all-in

   CFR technique: Pure CFR

   Iterations: 498 billion

   Strategy profile: current

   Translation: Hard, geometric similarity

2. A data biased response to aggregate data from all of the agents in the 2011 and 2012 heads-up no-limit ACPC events

   DBR $P_{\max}$: 0.25

   Abstraction: asymmetric

   Cards: IR KE-KO 169-9000-9000-3700

   Frequentist model cards: 1 bucket (ignores cards)

   Betting: 0.5, 0.75, 1, 1.5, 3, 6, 11, 20, or 40 times the pot size, or all-in

   CFR technique: external sampled

   Iterations: 116 billion

   Strategy profile: average

   Translation: Hard, geometric similarity

   Strategy notes: Hard translation based on geometric similarity was also used to build the frequentist model from the real game observations.

3. A data biased response based on aggregate data from agents that were not beaten by the 2013 Hyperborean TBR entry for at least 750 mbb/g

   DBR $P_{\max}$: 0.25

   Abstraction: asymmetric

   Cards: IR KE-KO 169-9000-9000-3700

   Frequentist model cards: 1 bucket (ignores cards)

   Betting: 0.5, 0.75, 1, 1.5, 3, 6, 11, 20, or 40 times the pot size, or all-in

   CFR technique: external sampled

   Iterations: 89 billion

   Strategy profile: average

   Translation: Hard, geometric similarity

   Strategy notes: Hard translation based on geometric similarity was also used to build the frequentist model from the real game observations.

Bankroll Instant Runoff

1. An abstract Nash equilibrium approximation using an asymmetric betting abstraction with more betting options for the opponent

   Abstraction: asymmetric

   Strategy cards: IR KE-KO 169-18630-3700-1175

   Strategy betting: 0.1, 0.25, 0.65, 1, 1.5, 3, 6, 11, or 20 times the pot size, or all-in

   Opponent cards: IR KE-KO 169-18630-3700-1175

   Opponent betting: min, 0.1, 0.25, 0.5, 1, 1.5, 3, 6, 11, or 20 times the pot size, or all-in

   CFR technique: Pure CFR

   Iterations: 82 billion

   Strategy profile: current

   Translation: Hard, geometric similarity

   Strategy notes: Both players are restricted in the timing and frequency of bets that are 1.5-pot or less. For bets shared by both positions, the opponent is strictly less restricted.