PRACTICAL STRATEGIC REASONING WITH APPLICATIONS IN MARKET GAMES

by

Patrick R. Jordan

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in The University of Michigan
2010

To Andrea, Carrie, Mom, and Dad.

Acknowledgments

I have been inspired and influenced by a great many people while writing my dissertation. I am profoundly indebted to all of the friends, family, and colleagues that have played a part in my time at the University of Michigan. Foremost, my advisor, Michael Wellman, provided indispensable guidance and patience—allowing me to jump from idea to idea until something fruitful emerged. I consider it an honor to have worked with someone so resolute in advancing science.

I would also like to thank the other members of my dissertation committee: Jeffrey MacKie-Mason, Satinder Singh Baveja, Rahul Sami, and David Parkes. Each provided insightful comments and criticisms that greatly improved the quality and direction of this thesis.

I have had a number of outstanding office mates, both as collaborators and friends. Christopher Kiekintveld was critical in the analysis of supply chains—a coauthor on numerous papers and a great teammate. Yevgeniy (Eugene) Vorobeychik collaborated on the analysis of profile search algorithms. Discussions with Chris and Eugene often produced deep insights about empirical game theory. Julian Schvartzman, along with Michael Wellman, introduced the strategy exploration problem. The Ad Auctions game would not have existed if not for the help of Lee Callender, Ben Cassell, Guha Balakrishnan, and Kemal Eren. Kevin Lochner and Daniel Reeves, to this day, selflessly subject themselves to my incessant questioning. Shih-Fen Cheng, Quang Duong, Yagil Engel, Jason Miller, Kevin O'Malley, Prateek Tandon, Bryce Wiedenbeck, and Craig Wright have been invaluable colleagues and friends.

I have always found support from a great group of present and former students, including Matthew Rudary, Erik Talvitie, Jonathan Sorg, Laura Fink, Kyla McMullen, Stefan Witwicki, James Boerkoel, Julie Weber, Dmitri Dolgov, Anya Osepayshvili, and William Rand.

I am very thankful for the generous help of the Computer Science and Engineering staff—in particular, Rita Rendell, Cindy Watts, Kelly Cormier, Jeanne Patterson, Karen Liska, and Dawn Freysinger. Collectively, they allowed me to focus on my academic pursuits, adeptly handling any administrative problems I manufactured.

The STIET (Socio-Technical Infrastructure for Electronic Transactions) fellowship and program was instrumental in developing my ideas for a multidisciplinary audience. The program was founded by Michael Wellman and Jeffrey MacKie-Mason and, most recently, directed by Yan Chen and managed by Karen Woollams.

Members of the Association for Trading Agent Research and participants in the Trading Agent Competition provided many of the strategies used in the ad auction and supply chain management analyses. Without their contributions, the empirical analysis contained in this thesis would not have been possible. As a participant in the competitions, I would like to thank David Pardoe and the TacTex team. During each tournament, the DeepMaize team spent a considerable amount of mental effort on counter intelligence—optimizing against predicted reactions by TacTex and the other teams. I believe this type of highly competitive environment greatly contributed to the realism of the scenario. In addition, Brown University's team, led by Amy Greenwald, provided valuable feedback early in the design process for the Ad Auctions game.

I would like to thank my friend and business partner, James Sweeney. James collaborated on many projects that allowed me to apply my academic pursuits to real-world business problems.

I would like to express my extreme gratitude to my family—my parents (Michael and Leslee Jordan), my sister (Carrie Jordan), and my grandparents (Richard and Kathleen Jordan; Donald and Nancy McLean; and Mary Julia McLean)—for years of encouragement and support, and to my great uncle, Father David Pepper, for steadfastly supporting the Irish of Notre Dame: *Go Blue*!

Finally, a special thanks to Andrea, to whom I am immeasurably grateful. Thank you for your ceaseless love and support, and for putting your dreams on hold for five years so that I could pursue mine.

Table of Contents

List of Tables

List of Figures

Abstract

Strategic reasoning is part of our everyday lives: we negotiate prices, bid in auctions, write contracts, and play games. We choose actions in these scenarios based on our preferences, and our beliefs about preferences of the other participants. Game theory provides a rich mathematical framework through which we can reason about the influence of these preferences. Clever abstractions allow us to predict the outcome of complex agent interactions, however, as the scenarios we model increase in complexity, the abstractions we use to enable classical game-theoretic analysis lose fidelity. In *empirical game-theoretic analysis*, we construct game models using empirical sources of knowledge—such as high-fidelity simulation. However, utilizing empirical knowledge introduces a host of different computational and statistical problems.

I investigate five main research problems that focus on efficient selection, estimation, and analysis of empirical game models. I introduce a flexible modeling approach, where we may construct multiple game-theoretic models from the same set of observations. I propose a principled methodology for comparing empirical game models and a family of algorithms that select a model from a set of candidates.

I develop algorithms for normal-form games that efficiently identify *formations*—sets of strategies that are closed under a (correlated) best-response correspondence. This aids in problems, such as finding Nash equilibria, that are key to analysis but hard to solve. I investigate policies for sequentially determining profiles to simulate, when constrained by a budget for simulation. Efficient policies allow modelers to analyze complex scenarios by evaluating a subset of the profiles. The policies I introduce outperform the existing policies in experiments.

I establish a principled methodology for evaluating strategies given an empirical game model. I employ this methodology in two case studies of market scenarios: first, a case study in supply chain management from the perspective of a strategy designer; then, a case study in Internet ad auctions from the perspective of a mechanism designer. As part of the latter analysis, I develop an ad-auctions scenario that captures several key strategic issues in this domain for the first time.

Chapter 1
Introduction

The Internet has brought about fundamental change to the world economy. The U.S. Census Bureau estimates that in 2007 the value of e-commerce shipments, sales, and revenues totaled \$4.33 trillion or approximately 15% of total economic activity.[1] Business-to-business transactions accounted for the lion's share of e-commerce activity (92.5%). Fueled by vast amounts of data and the increasing complexity of making business decisions in electronic markets, some firms now design complex software systems to execute key elements of their business strategies.

The prominence of Internet advertising has dramatically shifted marketing practices and serves as another example of the influence the Internet has had on the economy. In a recent survey,[2] the Interactive Advertising Bureau (IAB) reports a total of \$23.4 billion in annual Internet advertising revenue with search advertising accounting for 45%. To put this in perspective, consider that revenue from Internet advertising exceeded the revenue of traditional methods of advertising like broadcast (\$18.0B) and cable television (\$21.4B).

While this figure is staggering, it is the underlying technological paradigm shift that has caused the greatest excitement in related research communities. Unlike early forms of Internet advertising where contracts were negotiated in large blocks through the use of human sales teams, auctions are now the dominant process by which the allocation and pricing of advertisements is determined. Major search engines return paid advertisements alongside organic search results in response to user queries. Each query triggers a new

[1]See `http://www.census.gov/econ/estats/2007/2007reportfinal.pdf`
[2]See `http://www.iab.net/media/file/IAB_PwC_2008_full_year.pdf`

auction that allocates the ad slots to paying advertisers. In July of 2008, the top search engines received almost half a billion search queries per day.[3] Because of the sheer magnitude, search engines use highly optimized software systems to clear the auctions. These systems reason over statistical models of user and advertiser behavior while continuously adapting auction rules to optimize strategic goals. In addition to the software systems used by the search engines to implement the auctions, sophisticated advertisers employ software agents to manage their advertising campaigns. These agents may manage the daily bids of a few hundred to a few hundred thousand or more keywords. The agents act in semi-autonomous or fully autonomous fashion. I use the term *trading agents* to describe self-interested autonomous software agents that participate in markets.

This thesis addresses the problem of practical design and evaluation of complex scenarios involving multiple trading agents. These multiagent scenarios can be described as *games*. The goal of this thesis is to develop computational methodologies and techniques that enable modelers to apply *game-theoretic reasoning* to scenarios that defy classical analysis. Game theory provides a principled approach to designing and evaluating trading agents and the markets in which the agents participate.

1.1 Foundation

Research in understanding agent behavior has classically been considered an economic discipline. However, as markets and decisions that agents make participating in these markets increase in complexity, practical algorithmic and computational concerns arise. For instance, today's market designers must consider the algorithmic constraints on the complexity of the software required to facilitate their markets. Modern examples of these market mechanisms include the ad auctions of Google, Microsoft, and Yahoo; the supply networks of Wal-Mart that integrate directly with vendor inventory; electronic crossing net-

[3]See `http://searchenginewatch.com/3630718`

works (ECNs) of financial markets; and combinatorial auctions in which bidders can bid over multiple attributes and items.

From an agent's perspective, its strategic choices may be algorithmically complex as well. The environment the agent operates within may contain stochastic elements whose parameters and distributions must be learned. Therefore, reasoning under uncertainty is a core component of agent behavior. Firms may delegate business process decisions to trading agents, therefore understanding and evaluating the algorithmic behavior of trading agents is a primary concern of these firms and of the designers of the markets in which the firms interact.

Multiagent systems (MAS) research, a sub-discipline of artificial intelligence, has long relied on both theoretical and experimental analysis to understand the implications of alternative agent behaviors, or strategies. For situations amenable to analytic modeling, researchers often appeal to the framework of game theory (von Neumann and Morgenstern, 1944), primarily due to its generality and normative solutions.

For scenarios that are too complex or lack directly specified game forms (i.e., payoff or utility functions for a normal-form representation), analytic game theory is not immediately applicable. In such situations, *empirical game models* (Wellman, 2006), where observations or simulations of agent play are used to construct estimates of their utility, can support game-theoretic analysis despite lack of explicit game descriptions. Empirical game theory allows researchers to model a strategic scenario at a practical level of abstraction where analysis is computationally feasible and game-theoretic concepts can still be applied.

1.2 Empirical Game-Theoretic Framework

The concept of empirical games has antecedents in both the MAS and economics literature. In 1944, John von Neumann and Oskar Morgenstern introduced a framework for modeling scenarios involving multiple self-interested agents. Their theory of games pioneered a new economic discipline. Soon after, John Nash (1951) generalized this framework by extending von Neumann and Morgenstern's work on two-player zero-sum games. In 1967, John Harsanyi introduced *Bayesian games* to model situations where players are uncertain about the payoffs of other players. Many general models of games now exist that span a multitude of different modeling scenarios, however it is the non-cooperative games of Nash and the Bayesian games of Harsanyi that form the basis for the empirical game-theoretic framework used in this thesis.

Non-Cooperative Games

Nash (1951) investigated non-cooperative games consisting of a basic set of elements: *players*, *actions*, and *utilities*. Unless otherwise specified, I refer to the strategic or normal form.

Definition 1 (Strategic Game)**.** *A strategic game $\Gamma = \langle N, (A_i), (u_i) \rangle$ consists of*

- *a finite set of players, N, of size n indexed by i;*
- *a non-empty set of actions A_i for each player;*
- *a utility function $u_i : \times_{j \in N} A_j \to \mathbb{R}$ for each player.*

Following the notation of Osborne and Rubinstein (1994), I refer to $a = (a_j)_{j \in N}$ as an *outcome* or *joint action* and the set $A = \times_{j \in N} A_j$ as the *joint action space*. Each player's utility depends on the outcome chosen jointly by the players, not simply on its own action. In this setting, each player has full knowledge of its and the other players action sets and utility functions, respectively. This condition is termed *complete information*. Finally, each player chooses an action and all players choose their actions simultaneously.

Bayesian Games

In some scenarios, a player may be uncertain about the utility functions, actions, or the knowledge of other players (*incomplete information*) or may not know all of the actions that have been played (*imperfect information*). Harsanyi (1967) proposed a framework in which we transform a game of *incomplete information* into a game of *imperfect information* by including an additional player, commonly called *Nature*. In Harsanyi's model, Nature takes the first action by selecting its state. Each player has a Bayesian prior over the states and the priors are common knowledge. After Nature selects the state, each player chooses an action simultaneously. A player may condition the action it chooses upon a *signal* (also called a *type*), which is a function of the selected state of Nature. The utility received by the players is a function of the joint actions taken by all the players (including Nature).

Definition 2 (Bayesian Game (Osborne and Rubinstein, 1994))**.** *A* Bayesian game $\langle N, \Omega, (A_i), (T_i), (\tau_i), (p_i), (u_i)\rangle$ *consists of*

- *a finite set N of players;*
- *a set of states of Nature Ω;*

and for each player $i \in N$

- *a set of actions A_i;*
- *a set of signals T_i and a signal function $\tau_i : \Omega \to T_i$;*
- *a probability measure P_i on Ω for which $P_i(\tau_i^{-1}(t_i)) > 0$ for all $t_i \in T_i$;*
- *a utility function $u_i : \times_{j\in N} A_j \times \Omega \to \mathbb{R}$.*

A player's *strategy* is a mapping $s_i : T_i \to A_i$ where $a_i = s_i(t_i)$ and S_i is player i's strategy space. We refer to $s = (s_j)_{j\in N}$ as a *joint strategy* or *profile* and $S = \times_{j\in N} S_j$ as the *joint strategy space*. The expected (von Neumann and Morgenstern) utility of s to player i is

$$U_i(s) = \int_{\Omega} u_i(s(\omega), \omega)\, dP_i(\omega), \tag{1.1}$$

where $s(\omega) = (s_j(\tau_j(\omega)))_{j\in N}$. Harsanyi (1967) describes a Bayesian game in strategic form by using strategies in the place of actions with the corresponding expected utility, i.e., $\Gamma = \langle N, (S_i), (U_i)\rangle$. Unless otherwise stated, I use this formulation throughout this thesis to describe strategic scenarios.

Each profile s is associated with the set of neighboring profiles that can be reached through a unilateral deviation by a player.

Definition 3 (Unilateral Deviation Set)**.** *The* unilateral deviation set *for player i and profile*

$s \in S$ *is*

$$\mathcal{D}_i(s) = \{(\hat{s}_i, s_{-i}) : \hat{s}_i \in S_i\},$$

and the corresponding set unspecified by player is

$$\mathcal{D}(s) = \bigcup_{i \in N} \mathcal{D}_i(s).$$

Note that $s \in \mathcal{D}(s)$, and that $\hat{s} \in \mathcal{D}(s)$ implies $s \in \mathcal{D}(\hat{s})$. The notation ${}_s\mathcal{D}_{\hat{s}}$ is used to indicate that s and $\hat{s}$ are mutual deviations. Additionally, for $X \subseteq S$,

$$\mathcal{D}(X) = \bigcup_{s \in X} \mathcal{D}(s).$$

Let $\Delta(\cdot)$ represent the probability simplex over a set. A *mixed strategy* σ_i is a probability distribution over strategies in S_i, with $\sigma_i(s_i)$ denoting the probability player i will play strategy s_i. The *mixed strategy space* for player i is given by $\Delta_i = \Delta(S_i)$. Similarly, $\Delta^S = \times_{i \in N} \Delta_i$ is the *mixed profile space*.

Empirical Games

Like Bayesian games, empirical games model scenarios in which players may be uncertain about the utility of playing a profile. What distinguishes the Bayesian formulation from the empirical is that in an empirical game the utility functions are unknown. In the place of utility functions, we have a *simulator* that generates (stochastic) observations of utility from play.

In complex multiagent environments, strategies are often best described procedurally, in effect, as computer programs that take type information as input and return the selected action or sequence of actions as their output. For this reason, we often can do no better than treat these strategies as black boxes, analyzing them in terms of input-output but not internal structure. Thus a *simulator* is a (stochastic) function that maps joint strategies to outcomes, in the form of payoff vectors. Typically, a simulator will itself be realized as a program that generates type information and implements the interaction among the participating agents and the environment.

Definition 4 (Simulator)**.** *A* simulator *is a tuple* $\mathfrak{S} = \langle N, \Omega, P, (S_i), (\Pi_i)\rangle$ *consisting of*

- *a finite set of players* N *of size* n *indexed by* i*;*
- *a set of states* Ω*;*
- *a probability measure* P *on* Ω*;*

and for each player $i \in N$

- *a set* S_i *of strategies;*
- *a function* $\Pi_i : \times_{j \in N} S_j \times \Omega \to \mathbb{R}$ *corresponding to the payoff player* i *receives when a particular profile is simulated.*

Although the full strategy space allowed by the game simulator may be large, empirical game models usually restrict the strategy space to a small number of heuristically defined strategies (Walsh et al., 2002). Even within this restricted space, modeling accuracy is limited by statistical sampling, given the inherent stochastic behavior of typical simulation environments. Given a limited computational budget, modelers must choose carefully which strategy profiles to simulate and how many samples of those profiles to take.

One benefit of a simulation-based framework is that representations of a multiagent system can be made with varying degrees of fidelity to the underlying scenario. For instance, a stock market scenario may involve thousands of participants, each of whom are selecting from a potentially infinite number of trading strategies. A simulation of the scenario may be reduced to a manageable number of representative participants, each selecting from a handful of promising strategies. Depending on the computational budget, an analyst may increase or decrease the population size or available trading strategies in the simulator.

An individual run of the simulator, or *simulation*, produces an *observation* $\theta = (s, \pi)$, where s is the strategy profile simulated and $\pi = (\pi_1, \ldots, \pi_n)$ is the realization of the payoffs received by the players. Let $\Theta = \{\theta_k\}$ be the *observation set.* A simulator $\mathfrak{S}$ implicitly defines a strategic game over the set of players N and profile space S, but without an explicit utility function as would normally be specified in a game description. Instead, the observation set generated by the simulator provides the basis for an estimated game model, which is what I call the *empirical game*. I define empirical games below, but first I introduce a property called *gain preserving*.

Definition 5 (Gain Preserving)**.** *Let A and B be index sets and $f = (f_j)_{j \in B}$ be a vector-valued function such that $f_j : \mathbb{R}^{|A|} \to \mathbb{R}$. The function f is* gain preserving *if there exists a $\kappa > 0$ such that for every $\epsilon \geq 0$ and $x, y \in \mathbb{R}^{|A|}$, $\max_{i \in A}\ y_i - x_i \leq \epsilon \Rightarrow \max_{j \in B}\ f_j(y) - f_j(x) \leq \kappa\epsilon$.*

For a gain-preserving function f, I call κ the *gain-preserving constant*.

Definition 6 (Empirical Game)**.** *An* empirical game *is a tuple $\mathcal{E} = \langle \Gamma, \mathfrak{S}, \phi, \mu \rangle$ consisting of*

- *a strategic game $\Gamma = \langle N^{\Gamma}, S^{\Gamma}, u^{\Gamma} \rangle$;*
- *a simulator $\mathfrak{S} = \langle N^{\mathfrak{S}}, \Omega, P, S^{\mathfrak{S}}, \Pi^{\mathfrak{S}} \rangle$;*
- *a mapping $\phi : S^{\mathfrak{S}} \to S^{\Gamma}$ such that $\forall s^{\mathfrak{S}}, \hat{s}^{\mathfrak{S}} \in S^{\mathfrak{S}}$, ${}_{s}\mathcal{D}_{\hat{s}} \Rightarrow {}_{\phi(s)}\mathcal{D}_{\phi(\hat{s})}$;*
- *a gain-preserving mapping $\mu = (\mu_j)_{j \in N^{\mathfrak{S}}}$ where $\mu_j : \mathbb{R}^{|N^{\Gamma}|} \to \mathbb{R}$.*

In the empirical game $\langle \Gamma, \mathfrak{S}, \phi, \mu \rangle$, the strategic game Γ is termed the *embedded game* and μ_i associates the utilities in Γ to player i's payoff in $\mathfrak{S}$. Conceptually, the functions ϕ and μ allow us to move between the simulation space and the embedded game, while preserving game-theoretic interpretations of the scenario—such as the deviation relation. For every $i \in N^{\mathfrak{S}}$, I define player i's *empirical utility model* : $\tilde{u}_i^{\mathcal{E}}(s^{\mathfrak{S}}) = \mu_i(u^{\Gamma}(\ \phi(s^{\mathfrak{S}})\ )\ )$ for every $s^{\mathfrak{S}} \in S^{\mathfrak{S}}$. Note that as defined here, the embedded game need not employ the same player and profile space as the simulator. Because our evidence is statistical, we may choose a different Γ to model an empirical game depending on our set of observations. Determining which Γ is best is the one of the core research questions addressed in this thesis.

Consider the example two-player, four-action simulator whose mean payoffs are given in Table 1.1. The simulation strategy set for each player is $\{\mathrm{a}, \mathrm{b}\} \times \{\mathrm{A}, \mathrm{B}\}$. The total number of simulation profiles is 16. However, this example can be decomposed into two

	(a,A)	(b,A)	(a,B)	(b,B)
(a,A)	(6,6)	(8,3)	(6,6)	(8,3)
(b,A)	(3,8)	(4,4)	(3,8)	(4,4)
(a,B)	(6,6)	(8,3)	(6,6)	(8,3)
(b,B)	(3,8)	(4,4)	(3,8)	(4,4)

Table 1.1: Mean payoffs for a two-player, four-action simulator.

games that are additive in utility: the game over the first factor and the game over the second factor. Moreover, we find that the players' choices for the second factor are irrelevant (in mean utility). Therefore we can model the scenario using only the game over the first factor: $\Gamma = \langle\{\text{row}, \text{col}\}, \{a, b\}^2, u^\Gamma\rangle$. Let $s_{\text{row}} = (s^1_{\text{row}}, s^2_{\text{row}})$ be the strategy played by the row player and $s_{\text{col}} = (s^1_{\text{col}}, s^2_{\text{col}})$ be the strategy played by the column player. The function ϕ is given by $\phi(s_{\text{row}}, s_{\text{col}}) = (s^1_{\text{row}}, s^1_{\text{col}})$. The function $\mu = (\mu_i)_{i\in\{\text{row,col}\}}$, where $\mu_i(u_i, u_{-i}) = u_i$ and $i \in \{\text{row}, \text{col}\}$. The utility function of the embedded game is given in Table 1.2. Instead of having 16 profiles as in the simulation space, the embedded game has 4. This compact representation reduces the risk of *overfitting* the utilities in the embedded model—due to noise in the simulator.

	a	b
a	(6,6)	(8,3)
b	(3,8)	(4,4)

Table 1.2: An embedded utility function, u^Γ, for the simulated payoffs in Table 1.1.

Strategic Stability

One goal of game-theoretic analysis is to identify profiles that are strategically *stable*, that is, profiles where players have no incentive to unilaterally deviate. Stable profiles are a fundamental component of many of the analysis techniques developed in this thesis, however, approximately stable profiles are also of great value. I use a concept called *regret* to measure the stability of a profile. Before describing regret and the Nash equilibrium solution concept, I outline a supporting game-theoretic concept called the *best response*. The best-response correspondence is used in the regret calculation of a profile as well as calculating the regret of a set of profiles. I temporarily suppress the superscript Γ notation for ease of exposition.

Best-response correspondence

For a given player i, the *best-response correspondence*, $\mathcal{B}_i$, maps a given opponent profile σ_{-i} to the set of strategies that yield the maximum payoff, holding the other players' strategies constant. I define best-response correspondences for a set of profiles, as well as *overall best-response correspondences*, $\mathcal{B}$, that give the joint best-responses with respect to a profile or a set of profiles.

Definition 7 (Best Response)**.** *The player i best-response to opponent profile $\sigma_{-i} \in \Delta(S_{-i})$ is*

$$\mathcal{B}_i(\sigma_{-i}) = \underset{\hat{\sigma}_i \in \Delta_i}{\arg\max}\, u_i(\hat{\sigma}_i\ ,\ \sigma_{-i}),$$

and for $\Delta_{-i} \subseteq \Delta(S_{-i})$ is $\mathcal{B}_i(\Delta_{-i}) = \times_{\sigma_{-i} \in \Delta} \mathcal{B}_i(\sigma_{-i})$. The overall best-response *for profile $\sigma \in \Delta(S)$ is $\mathcal{B}(\sigma) = \times_{i \in N} \mathcal{B}_i(\sigma_{-i})$ and for $\Delta \subseteq \Delta(S)$ is $\mathcal{B}(\Delta) = \times_{\sigma \in \Delta} \mathcal{B}(\sigma)$.*

The symbols $\mathfrak{B}_i(\sigma_{-i})$, $\mathfrak{B}_i(\Delta_{-i})$, $\mathfrak{B}(\sigma_{-i})$, and $\mathfrak{B}(\Delta)$ denote to the pure-strategy variants of the best-responses—allowing for a slight abuse of notation:

$$\begin{aligned}
\mathfrak{B}_i(\sigma_{-i}) &= \mathcal{B}_i(\sigma_{-i}) \cap S_i \\
\mathfrak{B}_i(\Delta_{-i}) &= \mathcal{B}_i(\Delta_{-i}) \cap S_i \\
\mathfrak{B}(\sigma) &= \times_{i \in N} \mathfrak{B}_i(\sigma_{-i}) \\
\mathfrak{B}(\Delta) &= \bigcup_{\sigma \in \Delta} \mathfrak{B}(\sigma).
\end{aligned}$$

I introduce notation for the pure-strategy best-response to a set of profiles $X = \times_{i \in N} X_i$

where $\emptyset \subset X_i \subseteq S_i$:

$$\begin{aligned}
\mathfrak{B}_i(X_{-i}) &= \mathfrak{B}_i\left(\times_{j\in N\setminus\{i\}}\Delta\left(X_j\right)\right)\\
\mathfrak{B}_i^\dagger(X_{-i}) &= \mathfrak{B}_i\left(\Delta(X_{-i})\right)\\
\mathfrak{B}(X) &= \times_{i\in N}\mathfrak{B}_i(X_{-i})\\
\mathfrak{B}^\dagger(X) &= \times_{i\in N}\mathfrak{B}_i^\dagger(X_{-i})
\end{aligned}$$

The $\mathfrak{B}(\cdot)$ and $\mathfrak{B}^\dagger(\cdot)$ notation corresponds to differing independence assumptions on player i's *system of beliefs* (Bernheim, 1984). Under $\mathfrak{B}(\cdot)$, player i may not hold subjective beliefs of opponent correlation (opponents mix independently over their respective strategy set). Under $\mathfrak{B}^\dagger(\cdot)$, player i may hold subjective beliefs of opponent correlation.

Regret

The regret measures described in this section measure the stability of strategies and profiles, respectively.

Definition 8 (Strategy regret). *A player's* regret, $\delta_i(\sigma_i|\sigma_{-i})$, *for playing strategy* $\sigma_i \in \Delta_i$ *against opponent profile* $\sigma_{-i} \in \Delta(S_{-i})$ *is* $\max\limits_{s_i\in S_i} u_i(s_i,\sigma_{-i}) - u_i(\sigma_i,\sigma_{-i})$.

Note that this can equivalently be formulated as $u_i(\hat{\sigma}_i,\sigma_{-i}) - u_i(\sigma_i,\sigma_{-i})$ for any $\hat{\sigma}_i \in \mathcal{B}_i(\sigma_{-i})$. Given an efficient method for calculating a best-response, the latter formulation can speed up regret computations, especially if player i's strategy set is large or infinite.

Definition 9 (Maximum regret). *For strategy* $\sigma_i \in \Delta_i$, *the* maximum regret, $\delta_i^{\circledast}(\sigma_i)$, *is* $\max_{s\in S} u_i(s) - u_i(\sigma_i, s_{-i})$.

Savage (1954) uses a minimax decision criterion to select a strategy to minimize the maximum regret of the player. The maximum regret is related to the concept of approximate strategic dominance (Cheng and Wellman, 2007).

Definition 10 (δ-Dominance). *A strategy* $s_i^d \in S_i$ *is* δ-dominated *iff there exists a* $\sigma_i \in \Delta(S_i \setminus \{s_i^d\})$ *such that* $\delta \geq u_i(s_i^d, s_{-i}) - u_i(\sigma_i, s_{-i}), \forall s_{-i} \in S_{-i}$.

If the inequality is strict, we say that the strategy is *strictly* δ-dominated. If a strategy is not (strictly) δ-dominated, it is (strictly) δ-*dominant*. A strategy that has a maximum regret of δ is a δ-dominant strategy.

Finally, I use the regret of the constituent strategies to define the regret of a profile.

Definition 11 (Profile regret)**.** *The regret of profile $\sigma \in \Delta$, is the maximum gain from deviation from σ by any player. Formally, $\epsilon(\sigma) = \max_{i \in N} \delta_i(\sigma_i | \sigma_{-i})$.*

We can relate the regret of a profile in the embedded game to the regret of the profiles in the simulator—based on the empirical utility model. Let $\mathcal{E} = \langle \Gamma, \mathfrak{S}, \phi, \mu \rangle$, where κ is the gain-preserving constant for μ. Let $G = \langle N^{\mathfrak{S}}, S^{\mathfrak{S}}, \tilde{u}^{\mathcal{E}} \rangle$ be the strategic game formed from the simulation profile space and the empirical utility model. For every $\sigma^{\mathfrak{S}} \in \Delta^{\mathfrak{S}}$, let $\phi(\sigma^{\mathfrak{S}}) = \sigma^{\Gamma}$, where for each profile $s^{\Gamma} \in S^{\Gamma}$, the probability that the profile is played is given by $\sigma^{\Gamma}(s^{\Gamma}) = \sum_{s^{\mathfrak{S}} \in S^{\mathfrak{S}}} \phi(s^{\mathfrak{S}}) \mathcal{I}_{\{s^{\Gamma} = \phi(s^{\mathfrak{S}})\}}$ and $\mathcal{I}$ is the indicator function. Let $\epsilon(\cdot|\gamma)$ be the regret function with respect to game γ.

Theorem 1.2.1. *For every $\sigma^{\mathfrak{S}} \in \Delta^{S^{\mathfrak{S}}}$, $\epsilon(\sigma^{\mathfrak{S}}|G) \leq \kappa \cdot \epsilon(\phi(\sigma^{\mathfrak{S}})|\Gamma)$.*

Proof.

$$\begin{aligned}
\epsilon(\sigma^{\mathfrak{S}}|G) &= \max_{i \in N^{\mathfrak{S}}} \max_{s_i^{\mathfrak{S}} \in S_i^{\mathfrak{S}}} \tilde{u}_i^{\mathcal{E}}(s_i^{\mathfrak{S}}, \sigma_{-i}^{\mathfrak{S}}) - \tilde{u}_i^{\mathcal{E}}(\sigma^{\mathfrak{S}}) \\
&\leq \kappa \max_{j \in N^{\Gamma}} \max_{s_j^{\Gamma} \in S_j^{\Gamma}} u_j^{\Gamma}(s_j^{\Gamma}, \phi(\sigma^{\mathfrak{S}})_{-j}) - u_j^{\Gamma}(\phi(\sigma^{\mathfrak{S}})) \\
&= \kappa \cdot \epsilon(\phi(s^{\mathfrak{S}})|\Gamma).
\end{aligned}$$

□

Nash and Bayes-Nash equilibrium

The equilibrium concept proposed by Nash (1951) for non-cooperative games remains the foremost solution concept in game theory. Nash proved that such an equilibrium always exists for finite games.

Definition 12 (Nash equilibrium)**.** *A* Nash equilibrium *(NE) is a profile $\sigma \in \Delta^{\Gamma}$ such that $\sigma \in \mathcal{B}(\sigma)$, therefore $\epsilon(\sigma) = 0$.*

Nash defined this equilibrium over the set of actions in a non-cooperative game. Harsanyi (1967) extended the Nash equilibrium to Bayesian games by constructing a strategic game where the actions are the strategies of the Bayesian game and the utilities of the strategic game are given by the expected utilities of the strategy profiles, i.e, $\langle N, (S_i), (U_i)\rangle$. Harsanyi's solution concept is known as a *Bayes-Nash equilibrium*, however I simply refer to it as a *Nash equilibrium* when discussing empirical games.

Corollary 1.2.2. *If $\sigma^{\Gamma} \in \Delta^{S^{\Gamma}}$ is a Nash equilibrium in Γ and, for some $\sigma^{\mathfrak{G}} \in \Delta^{S^{\mathfrak{G}}}$, $\phi(\sigma^{\mathfrak{G}}) = \sigma^{\Gamma}$, then $\sigma^{\mathfrak{G}}$ is a Nash equilibrium in $\langle N^{\mathfrak{G}}, S^{\mathfrak{G}}, \tilde{u}^{\mathcal{E}}\rangle$.*

Proof. Follows directly from Theorem 1.2.1. $\square$

Approximate equilibria may be of interest in general, and are especially salient when analyzing pure profiles for games that may not exhibit pure-strategy Nash equilibria. We say that profile σ is an ϵ-*Nash equilibrium*, when the profile regret of σ is ϵ.

Constrained strategic equilibrium

Armantier et al. (2008) defined the concept of *constrained strategic equilibrium* (CSE) as an approximation of Bayesian Nash equilibrium (BNE) induced by constraints on the strategy space. Let $S^* \subseteq S$ be the BNEs of the game $\langle N, (S_i), (u_i)\rangle$. Let $S^k \subseteq S$ be the constrained strategy set at index k and S^{k*} be the Bayesian Nash equilibria of the constrained game $\langle N, (S_i^k), (u_i)\rangle$. An element of S^{k*} is a CSE. Assume $\forall k > 0,\ S^k \subset S^{k+1}$.

Theorem 1.2.3 (Corollary 2.2 of Armantier et al. (2008))**.** *If S is compact, u is continuous and there exists CSE s^{k*} $\forall k > 0$, then there exists a BNE in S, and any sequence of CSEs $\{s^{k*}\}_{k=1}^{\infty}$ has a subsequence that converges towards a BNE.*

Strategy sets in empirical games will generally not conform to Armantier et al.'s compactness requirement; nevertheless, their result provides theoretical support for our expectation that equilibria in empirical games will improve in approximation to equilibria of the true game as more strategies are evaluated.

1.3 Market Games and the Trading Agent Competition

Since 2000, the Trading Agent Competition (TAC) series of tournaments has spurred researchers to develop improved automated bidding techniques for an array of challenging market domains.[4] The original TAC game presented a travel-shopping scenario; subsequent games have addressed problems in supply chain management, market design, and ad auctions. By continually introducing new games, the TAC series engages the community in an expanded set of strategic issues bearing on trading agent design and analysis. A key feature of these games is that—like most realistic market environments—they are sufficiently complicated (severely imperfect and incomplete information revealed over time throughout dynamic activity) to defy analytic solution. Thus, empirical methods appear indispensable to progress. The TAC/SCM and TAC/AA games provide an experimental testbed for many of the techniques described in this thesis, while TAC/Travel makes less frequent appearances. Short descriptions of each game follow. In addition, each game has a detailed specification available.

TAC/Travel

TAC/Travel (Wellman et al., 2007) is an eight-player travel-shopping scenario where each player acts as a travel agent, with the goal of assembling travel packages. Each agent acts on behalf of eight clients, who express their preferences for various aspects of the trip. The objective of the travel agent is to maximize the total satisfaction of its clients (the sum of the client utilities) minus expenditures. An annual competition for the game was run every year from 2000–2006. Over 70 entries have competed over the entire history of the yearly competitions, with many teams submitting entries over the course of multiple competitions. There are over 30 scholarly publications, including one book, that analyze the entries and the game itself.

[4]See `http://tradingagents.org`, and `http://www.sics.se/tac`.

TAC/SCM

TAC/SCM (Arunachalam and Sadeh, 2005) is a six-player supply chain management scenario where players act as manufacturers, who must compete with each other for both supplies and customers, and manage inventories and production facilities. The objective of each manufacturer is to maximize its accumulated profits over the course of a simulated year. An annual competition for the game has run every year since 2003. Participants have created over 150 entries and over 50 corresponding scholarly publications.

TAC/Market Design

TAC/Market Design (Niu et al., 2008) is a variable-player double auction market scenario where agents act as market specialists. Players in the game create respective marketplaces that facilitate transactions between potential buyers and sellers currently associated with the marketplace. Players compete over three objectives: profit share from fees, market share, and transaction efficiency. The corresponding competition has run every year since 2007. Participants have created over 30 entries and at least 10 corresponding publications.

TAC/AA

TAC/AA (Chapter 9) is an eight-player sponsored search advertising scenario where agents act as advertisers. Players manage their respective ad campaign by selecting bids and ads to be displayed for each day. The objective of each advertiser is to maximize its accumulated sales profits net advertising expense over the course of a simulated 2-month campaign. Participants created 15 entries for the July 2009 inaugural competition.

1.4 Thesis Overview

In this thesis, I investigate a general methodology for analyzing complex strategic scenarios with empirical game-theoretic models. In doing so, I consolidate a collection of publications in which I, together with my coauthors, focus on five key research questions.

Question 1. *Can we reduce the variance in our estimates of the parameters of an empirical game model?*

Chapter 2 covers Monte Carlo methods for efficiently estimating parameters of the utility function of the embedded game. These methods focus on reducing the variance of unbiased estimators of the mean utility. Many of the standard variance reduction techniques originating from the simulation and control literature are applicable to multiagent simulation. I review these methods as well as the advantage-sum technique that has been used to analyze the performance of computer poker agents. The advantage-sum technique bears a strong resemblance to the method of control variates. New to this thesis, I introduce a family of control variate methods that extend the approach used by Wellman et al. (2007) and Jordan et al. (2007) to reduce the variance of the payoff estimates from the TAC/Travel and TAC/SCM simulators, respectively.

Question 2. *Given an empirical model of the game, how should strategies be evaluated?*

Chapter 3 explores and extends a regret-based approach to evaluating strategies in an empirical game model (Jordan et al., 2007). The NE-regret evaluation measure induces a rational ranking over each player's strategies. This ranking can be used to align the incentives of the entrants in research competitions with the goals of the competition designers.

Question 3. *Given an empirical model of the game, can we efficiently find (minimal) sets of strategies that are closed under rational behavior?*

Strategy sets *closed under rational behavior* are sets that are closed under a best-response correspondence. Chapter 4 introduces algorithms to find closed and approximately closed strategy sets by extending the approach of Benisch et al. (2006). Strategy sets closed under rational behavior are useful for determining restricted games—games whose strategy sets are restricted with respect to the base game—that capture relevant strategic information about profiles and strategies, such as regret, in the base game. The proposed algorithms are an improvement over the related iterated weaker-than-weak dominance strategy pruning algorithms of Cheng and Wellman (2007) on the tested market games.

Question 4. *Given a set of heuristic strategies and a computational budget for simulating agent play, how should modelers optimally select profiles to be simulated?*

Chapter 6 introduces the profile selection problem and categorizes the existing approaches by the payoff observation model they assume. Under each payoff model, I experimentally evaluate the known approaches from prior literature along with new algorithms and variants, on a range of game instances and game classes. In each case, I find that the new algorithms outperform the existing algorithms.

Chapter 7 applies the methods of Chapter 4 to the strategy exploration problem defined by Schvartzman and Wellman (2009a). I introduce a new method that attempts to form approximately-closed strategy sets in a minimal number of steps. This new method outperforms the existing policies in the experiments.

Question 5. *Given observations of agent play, how should empirical models of the underlying scenario be evaluated?*

In Chapter 8, I introduce a flexible approach where we may construct multiple game-theoretic models from the same set of observations. This approach gives a principled methodology for selecting an empirical model for a given scenario and set of observations. I propose a family of heuristic search algorithms that select an empirical game model from a set of candidates.

Chapters 2–4 and 6–8 form the basis for the empirical game-theoretic analysis framework. Chapters 5 and 9 employ this framework in two case studies: supply chain management and Internet ad auctions. The case study in supply chain management focuses on the design and evaluation of SCM strategies, in particular the design process used for DeepMaize. Additionally, I compare the progress of agents as a whole over subsequent tournament years.

The case study in Internet ad auctions focuses on the design and evaluation of ad auction mechanisms. As part of this thesis, I introduce a representative simulator that extends existing models in the literature. This simulator was used in the TAC/AA 2009 tournament. Fifteen teams participated in the tournament. Using the participants' strategies, I optimize the auction mechanisms within the simulated environment.

Chapter 2
Estimating Parameters

In this chapter I describe methods to efficiently estimate parameters of the utility function, u^Γ, of the embedded game Γ. The utility function may have many different parameters—for instance, a parameter specifying the mean utility for each player in each profile. The parameters must be estimated from observations of agent play. To this end, I review the existing body of literature on Monte Carlo methods for parameter estimation and describe how they can be employed when modeling empirical games.

In order to facilitate game-theoretic reasoning, we need an estimate for the mean payoff, $u_i(s)$, for each player in each profile. In general, if Γ is written in normal form and each player has at most P strategies to choose from, we need $\mathcal{O}(nP^n)$ such estimates. The realized payoffs from simulation are a function of strategic choices made by the players and stochasticity in the environment. With enough samples, the random effects of the environment are simply averaged away. However, simulating the market games described in this thesis is costly and observations are dear. Therefore, *efficient* parameter estimation methods are of great importance.

L'Ecuyer (1994) and Ross (2001) describe several important variance reduction techniques from the simulation research literature. Section 2.1 reviews four of these methods and highlights important aspects as they relate to multiagent simulation. Three of the four methods manipulate the stochastic inputs to the simulator and therefore require a simulation framework conducive to the respective type of manipulation. The fourth, *the method of control variates*, exploits the correlation between the stochastic inputs and the payoffs.

This method can be applied whenever the stochastic inputs are observable and requires no special consideration on the part of the simulation framework designer.

2.1 Variance Reduction in Monte Carlo Methods

Metropolis (1987) reflects on the early days of digital computing circa 1945 and a new computation technique from that era: *the Monte Carlo method.* The method originated out of a need to estimate parameters of complex physical systems (Metropolis and Ulam, 1949). At that time, this meant calculations related to the development of the hydrogen bomb on the newly introduced ENIAC. Since their inception, applications of Monte Carlo methods have expanded to include many challenging problems in various engineering and scientific disciplines.

Hammersley and Handscomb (1964) describe Monte Carlo methods as a branch of mathematics concerned with experiments on random numbers. They are particularly effective when estimating parameters of uncertain systems that are too complex or otherwise expensive to be ascertained analytically. At their core, Monte Carlo methods use repeated simulation to construct estimates of the parameters in question.

L'Ecuyer (1994) provides an extensive review of Monte Carlo methods that forms the basis of this section. However, because we are modeling games, there are added complications. For example, with the exception of the *common random variables method*, L'Ecuyer considers parameter estimation of a single "system" of interest. This corresponds to a single player's mean payoff estimate in a single profile. Instead, we are interested in a family of payoff estimates, one for each player in each profile. Moreover, the relationship between the payoff estimates for deviating profiles plays a central role in our analysis. I proceed by introducing the framework as described by L'Ecuyer, making exception where necessary to incorporate multiagent considerations.

We are interested in some statistical parameter π. Through simulation, we compute the realization of Z, an estimator of π. For instance, Z may be the sample average of several simulation runs. Thus, Z is the result of the entire simulation process, which may involve multiple runs through a simulator. The estimator is defined over probability space $(\Omega, \mathcal{B}, P)$ and is a measurable function of $\omega \in \Omega$, where Ω is the set of states of Nature, $\mathcal{B}$ is a σ-algebra (typically a Borel algebra), and P is a probability measure. Therefore $Z = h(\omega)$ for some h. Nelson (1985) describes the function h by three transformations such that $h(\omega) = T_3(T_2(T_1(\omega)))$. The (multivariate) random variable X, where $X = T_1(\omega)$, is considered the *input* to the simulator. The *output*, Y, of the simulator is a transformation of the input given by $Y = T_2(X)$. The final transformation, $Z = T_3(Y)$, gives the performance statistics for the simulation.

In this section I consider a simple motivating example. We have a single player with a single strategy. During the course of simulation a single random variable X is generated. The variable X is normally distributed. The payoff, Π, to the player is a measurable function of X given by $\Pi = g(X)$. Figure 2.1 shows $g(x)$ for the example. Our parameter of interest, π, is the expectation of Π. In this example, $g(x)$ is a step function that pays off 100 if X is greater than the threshold of two. Therefore, we can calculate $\pi = 100 \cdot (1 - \Phi^{-1}(2)) \approx 2.275$, where $\Phi^{-1}(\cdot)$ is the inverse standard normal cumulative distribution function. Using L'Ecuyer's notation, we devise an experiment where we create m samples of the input, therefore $X = (x_1, \ldots, x_m)$. For each of the sample inputs, we calculate the simulation output, in aggregate given by $Y = (g(x_1), \ldots, g(x_m))$. The performance statistic, Z, is the sample average of simulation output: $Z = \frac{1}{m} \sum_{i=1}^{m} g(x_i)$. Notice that $\mathbb{E}[Z] = \pi$, therefore Z is an unbiased estimator for π. I call this the *vanilla* Monte Carlo method and use the symbol $\bar{\Pi}$ to represent this specific definition of Z. I use alternate forms of the statistic Z in subsequent methods.

Figure 2.2 shows a trace of a 1,000-sample vanilla Monte Carlo estimation of π. The dark line shows $\bar{\Pi}$ and the gray lines show 95% confidence intervals for π. The confidence intervals are fairly wide, even after 1,000 samples. This is because the region where the

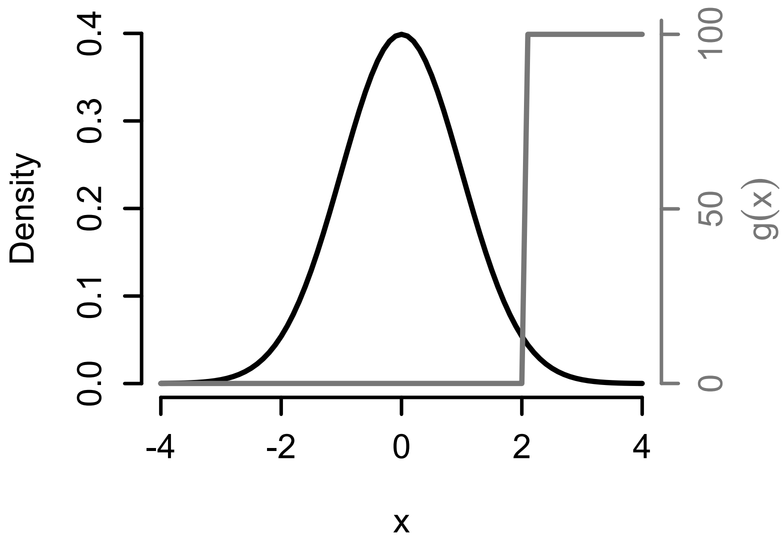


Figure 2.1: Single player, single action simulator

payoff is 100 has a low probability of occurrence.

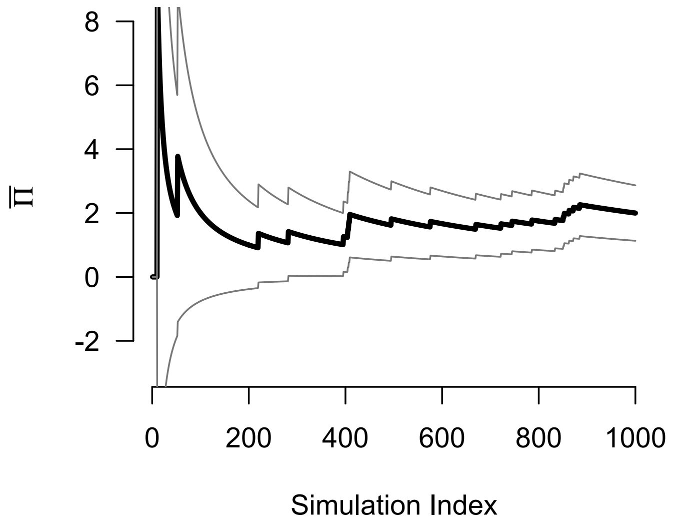


Figure 2.2: A 1,000 sample trace of the simple example.

Efficient Monte Carlo estimation considers the tradeoff between the computational cost of simulation and the level of precision for parameter estimates. In the approach of Nelson (1985), transformations T_1, T_2, and T_3 determine the *mean square error* (MSE) of the statistic Z given by

$$\mathrm{MSE}[Z] = \int_{\Omega} [h(\omega) - \pi]^2 dP(\omega). \tag{2.1}$$

When Z is an unbiased estimator of π, the mean square is equivalent to the variance of Z. *Efficiency* in Monte Carlo estimation is inversely proportional to the product of the mean square error and the expected cost of computing Z. Therefore, if we have two performance statistics, both with the same expected cost, we would prefer the one with smaller mean square error.

Wilson (1984) uses the term *variance reduction* to describe this process. He lists two broad categories for *variance reduction techniques* (VRTs): *correlation methods* and *importance methods*. Correlation methods exploit correlations in the random variables to reduce variance. Such methods include *antithetic variables*, *common random variables*, and *control variates*. Importance methods transform the input in order to focus sampling on pivotal regions of the sample space. Such methods include the method of *importance sampling* (described in this section) and additional methods like *stratification* (L'Ecuyer, 1994). Nelson (1985) takes a more elemental view of variance reduction, in which VRTs are formed by modifying the three basic transformations. In the remainder of this section, I review methods of antithetic variables, importance sampling, common random variables, and control variates, as they relate to parameter estimation in empirical games.

Antithetic Variables

Suppose we have two unbiased estimators, Z and Z', of π. We can combine these estimators into a single, unbiased estimator, $\hat{Z}$, by averaging Z and Z'. The variance of $\hat{Z}$ is

$$\mathrm{Var}[\hat{Z}] = \frac{\mathrm{Var}[Z]}{4} + \frac{\mathrm{Var}[Z']}{4} + \frac{\mathrm{Cov}[Z, Z']}{2}.$$

Suppose Z and Z' are generated independently with equal variance. The covariance term would vanish and we would be left with $\mathrm{Var}[\hat{Z}] = \mathrm{Var}[Z]/2$. In our simple example, this is analogous to the variance achieved through a vanilla Monte Carlo approach using $2m$ samples of the input, instead of m. What if, however, we could find Z and Z' with equal variance, but negative covariance? We would achieve a variance reduction.

Suppose in our simple example, $\omega \equiv \{\chi_k \sim U[0,1] \mid 1 \leq k \leq m\}$ and $X(\omega) = (\Phi^{-1}(\chi_1), \ldots, \Phi^{-1}(\chi_m))$. Consider the input vector $X(1-\omega)$ where $1-\omega \equiv \{1-\chi_k \mid \chi_k \in \omega\}$. Here, $1-\omega$ is the *antithetic sequence*. In the simple example, $X(\omega) = -X(1-\omega)$. If, for some k, $x_k(\omega) > 2$, then $x_k(1-\omega) < 2$. Therefore, $Y(\omega)$ and $Y(1-\omega)$ have the same variance, but are negatively correlated. Similarly, $Z(\omega)$ and $Z(1-\omega)$, which are the averages of $Y(\omega)$ and $Y(1-\omega)$, respectively, have negative covariance.

Let $\hat{Z}(\omega) = \frac{1}{2}(Z(\omega) + Z(1-\omega))$. The relationship between the variance of Z and $\hat{Z}$ is $\mathrm{Var}[\hat{Z}] < \mathrm{Var}[Z]/2$. In other words, for the same number of sample points, we have an estimator with smaller variance than the original. Figure 2.3 shows a trace of a 1,000-sample Monte Carlo estimation of π using antithetic variables. Because each step of the simulation uses two simulation observations, I report indices only up to 500.

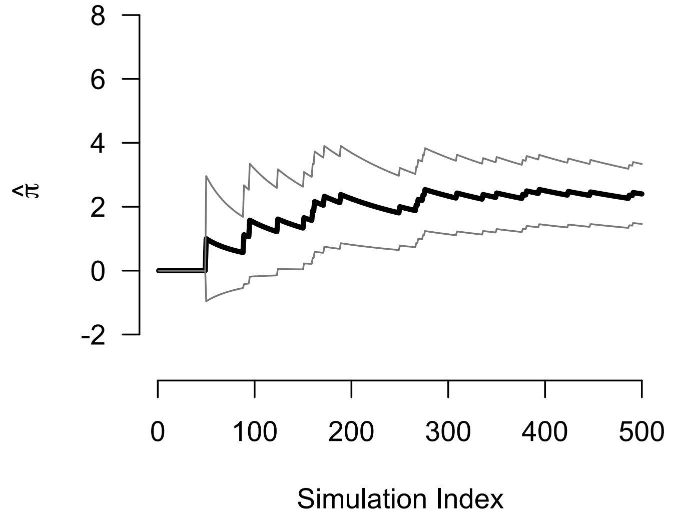


Figure 2.3: A 1,000-sample trace of the simple example using antithetic variables.

The effect of the antithetic variables is only slight in this example, because the negative correlation is weak (anti-correlated observations occur with low probability). Chapter 9 provides an example of antithetic variable analysis, through the use of capacity assignments in the TAC/AA tournament, where the negative correlation is significant.

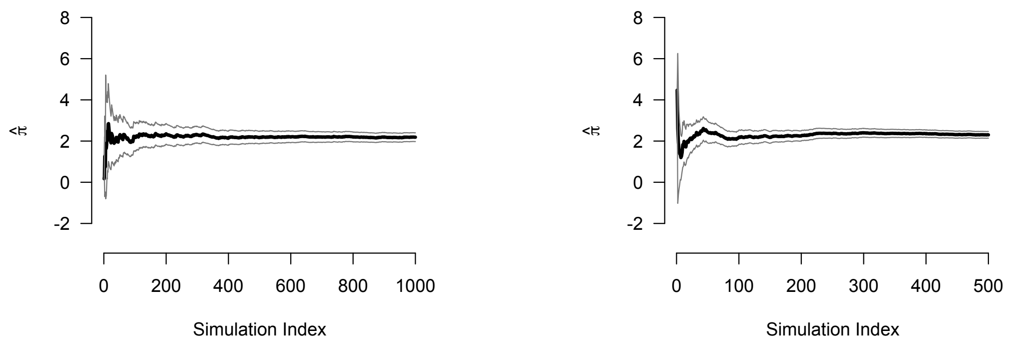
Importance Sampling

The *method of importance sampling* is particularly useful when rare events have a significant impact on parameter estimation, as is the case in our simple example. Importance sampling reduces variance by transforming the simulation input and focusing simulation on regions that are "important". The input is transformed from the original probability measure, P, to a new measure, Q. However, we adjust the resultant output, $h(\omega)$ under the Q-measure, such that the expectation of the output is equal to the expectation of $h(\omega)$ under the P-measure. This is done through the use of the Radon-Nikodym derivative, alternatively called the *likelihood ratio*:

$$\begin{aligned}
\mathbb{E}_P[h(\omega)] &= \int_\Omega h(\omega) dP(\omega) \\
&= \int_\Omega \left[h(\omega)\left(dP/dQ\right)(\omega)\right] dQ(\omega) \\
&= \mathbb{E}_Q[h(\omega)L(P,Q,\omega)]
\end{aligned}$$

where $L(P,Q,\omega)$ is the Radon-Nikodym derivative. We choose a Q-measure as to minimize $\int_\Omega [h(\omega)L(P,Q,\omega)-\pi]^2 dQ(\omega)$.

In our simple example, we can select Q such that X is still normally distributed, but centered around 2 instead of 0. Under this Q-measure, we are equally likely to observe a high or low value from $u(\cdot)$. However, we have to adjust the observation by the likelihood it would be observed under the P-measure. This likelihood ratio is the density of the standard normal divided by the density of a unit normal with mean two both evaluated at x. Figure 2.4(a) shows a trace of a 1,000-sample Monte Carlo estimation of π using only importance sampling. Because of the rare event nature of the simple example, the efficiency of this method is a vast improvement over the vanilla method and the antithetic variable method. However, we can combine methods to further improve efficiency. Figure 2.4(b) shows a trace of a 1,000-sample Monte Carlo estimation of π using importance sampling and antithetic variables, which is an improvement over importance sampling alone.



(a) Without antithetic variables (b) With antithetic variables

Figure 2.4: A 1,000-sample trace of the simple example using importance sampling.

Common Random Variables

Thus far, I have considered methods that efficiently calculate parameter estimates for a single $u_i(s)$. Now I consider the more general case where we are interested in estimating the differences in mean payoffs. Computing differences in payoffs is fundamental to regret calculations.

Suppose π_1 and π_2 are two unknown mean payoffs for two different strategies, estimated by Z_1 and Z_2, respectively. Let $Z_\delta = Z_1 - Z_2$ and suppose $\mathbb{E}[Z_\delta] = \pi_1 - \pi_2$. The variance of Z_δ is

$$\text{Var}[Z_\delta] = \text{Var}[Z_1] + \text{Var}[Z_2] - 2\text{Cov}[Z_1, Z_2].$$

Like with antithetic variables, the covariance term vanishes if Z_1 and Z_2 are generated independently. However, if we can induce a positive covariance, then we can decrease the variance of Z_δ. L'Ecuyer (1994) describes a method in which the same random numbers are used in each simulation experiment. Similar to the antithetic variables method, we use $Z_1(\omega)$ and $Z_2(\omega)$, with the same ω, to compute Z_δ. Care must be taken so that the same random numbers are used in the exact same "spot" in both scenarios.

In general, this design process is called *synchronization* in the simulation literature. Achieving synchronization is not a trivial task in the market games listed in Chapter 1. Sodomka et al. (2007) and Collins and Ketter (2009) make notable progress in achieving synchronization in TAC/SCM by modifying the standard simulator.[1] The authors' primary motivation for designing the synchronized systems is to improve statistical significance in paired t-tests, however we can also exploit their synchronization efforts for empirical game-theoretic analysis.

The Method of Control Variates

In complex market simulations such as TAC/SCM and TAC/AA, observable environmental variables may exist that correlate well the payoffs agents receive. The method of control variates exploits correlations between random variables whose expectations are known, called *control variates* or *control variables*, and a statistical estimator of interest. Unlike methods described above, control variates do not require manipulation of simulation inputs. This allows us to perform after-the-fact reductions in variance, a particularly convenient property when we do not control the path of simulation.

Lavenberg and Welch (1981) and Lavenberg et al. (1982) provide excellent general references for the method of control variates. I review their approach as it applies to multiagent simulations, using similar notation for convenience. I start with a single-player, single-strategy setting, and then incorporate multiple players and strategies into the final model.

[1]Each market factor is generated by a repeatable pseudo-random sequence. For example, the supply market factors are generated by a mean reverting random walk. Sodomka et al. (2007) modify the server by exposing seed parameters for each sequence.

As above, let π (the expected payoff) be our desired statistical parameter and Π a sample payoff. Hence, Π is an unbiased estimator of π from one simulation run. Let $C = (C_1, \ldots, C_Q)^T$ be a vector of Q control variables, such that C_q is generated from the same simulation run as Π, has a known expectation, and is correlated with Π. Let $\mu_C = \mathbb{E}[C]$ and $\mu_q = \mathbb{E}[\mu_q]$ for $q \in \{1, \ldots, Q\}$. For any constant vector α, $\Pi^C(\alpha) = \Pi - \alpha^T(C - \mu_C)$ is an unbiased estimator of π. Let α^* minimize the variance of that estimator, in particular,

$$\alpha^* = \Sigma_C^{-1} \sigma_{\pi,C},$$

where Σ_C is the covariance matrix of C and $\sigma_{\pi,C} = (\mathrm{Cov}[\Pi, C_1], \ldots, \mathrm{Cov}[\Pi, C_Q])^T$. At α^*, the variance of this estimator is related to the variance of π by

$$\frac{\mathrm{Var}[\Pi^C(\alpha^*)]}{\mathrm{Var}[\Pi]} = 1 - R^2_{\Pi,C},$$

where

$$R^2_{\Pi,C} = \left(\frac{1}{\mathrm{Var}[\Pi]}\right) \sigma^T_{\pi,C} \left(\Sigma_C^{-1}\right) \sigma_{\Pi,C}.$$

In practice $\sigma_{\Pi,C}$ and Σ_C are often not known, forcing us to estimate α^*. Assume we have K independent samples from simulation. To estimate α^*, Lavenberg et al. propose a regression of the form

$$\pi = \mu + \alpha^T(C - \mu_C) + e.$$

Let $\hat{\alpha}$ be the estimate for α^* obtained by the regression. If $(\Pi, C_1, \ldots, C_Q)$ is multivariate normal and $K > Q + 2$, then

$$\frac{\mathrm{Var}[\Pi^C(\hat{\alpha})]}{\mathrm{Var}[\Pi]} = \left(\frac{K-2}{K-Q-2}\right)(1 - R^2_{\pi,C}).$$

L'Ecuyer (1994) observes that, when the multinormality assumption is violated, the estimator is generally biased and may have larger variance than Π for small values of K.

However, asymptotically there is no loss in estimating α^*. Additionally, Lavenberg and Welch (1981) and Lavenberg et al. (1982) report that, in experiments, the *ratio* is relatively accurate, even when the normality assumption is violated. Therefore we have a quantifiable heuristic for selecting the number of control variables relative to the number of samples.

Lavenberg et al. developed the method of control variates in a simulation domain in which there is a single parameter of interest. In multiagent scenarios, we have a *set* of parameters of interest. A single run of the simulator may reveal information relevant only to a subset of these parameters, namely, the parameters corresponding to the profile being simulated. Depending on the observations available to us, we try various methods, taking the view that any reduction in variance is welcome. I consider the following variations on the method of control variates:

- **Luck-only:** the effect of the control variables on mean payoff is the same for all strategies, regardless of the opponent profile;
- **Strategy level:** the effect of the control variables on mean payoff is the same for a given strategy, regardless of the opponent profile;
- **Profile level:** the effect of the control variables on mean payoff is possibly dependent on the given strategy and the opponent profile.

Choosing the best method involves reasoning about the relationship between the observation data set and empirical game model. In order to apply the *strategy* and *profile-level* methods, we need to have a reasonable number of observations for each strategy and profile, respectively. When the set of strategies under consideration is small, this assumption may make sense. However, as is the case with the empirical games modeled in this thesis, it is unrealistic to assume that we can sample every possible profile a significant number of times. I review each model in turn.

Profile-level methods

In the *profile-level method*, we can straightforwardly follow Lavenberg et al.'s approach and calculate control variates independently for each player in each profile. Therefore, we create a statistic

$$\Pi_i^{s,C}(\alpha) = \Pi_i^s - \alpha^T(C - \mu_C) \tag{2.2}$$

for every $i \in N$ and $s \in S$. The control variables C can be unique to each player and profile. To estimate α^*, we solve a regression of the form

$$\Pi_i^s = \mu_i^s + \alpha^T(C - \mu_C) + e.$$

The number of regressions we perform is exponential in the number of players and strategies, thus requiring an exponentially large data set to regress on.

Strategy-level methods

In the strategy-level method, the control variables C and coefficients α are the same for every strategy. Therefore, the distribution over opponent profiles in the training set is very influential. One way to reduce the influence of the opponent profiles is to introduce a factor variable for each opponent profile. To estimate α^*, we solve a regression of one of the following forms:

$$\begin{aligned}
\pi_i^{s_i} &= \mu_i^{s_i} & &+\alpha^T(C - \mu_C) + e, \\
\pi_i^{s_i} &= \mu_i^{s_i} + \sum \mu_{s_{-i}} \mathcal{I}_{\{s_{-i}\}} & &+\alpha^T(C - \mu_C) + e,
\end{aligned}$$

where $\mathcal{I}_{\{x\}}$ is an indicator variable that is one if x occurs in the profile and 0 otherwise. In the first regression, the variance is explained solely by the control variates. This latter regression works well if the payoffs from simulation are additively decomposable into strategic factors and *luck* factors. The number of regressions we perform is linear in the number of strategies—an improvement over the profile-level method.

Luck-only methods

In the luck-only method, the control variables C and coefficients α are the same for every profile. To estimate α^*, we solve a regression of one of the following forms:

$$\begin{aligned}
\pi_i &= \mu_i & &+\alpha^T(C-\mu_C)+e,\\
\pi_i &= \mu_i+\sum\mu_{s_i}\mathcal{I}_{\{s_i\}} & &+\alpha^T(C-\mu_C)+e,\\
\pi_i &= \mu_i+\sum\mu_{s_{-i}}\mathcal{I}_{\{s_{-i}\}} & &+\alpha^T(C-\mu_C)+e,\\
\pi_i &= \mu_i+\sum\mu_{s}\mathcal{I}_{\{s\}} & &+\alpha^T(C-\mu_C)+e.
\end{aligned}$$

The various forms of regression correspond to our assumptions on how payoffs are (approximately) additively decomposable. In the first regression, the variance is explained solely by the control variates, whereas in the latter forms the variance is explained by a combination of the control variates and strategy, opponent profile, and profile factors, respectively.

Exploiting ex-interim form

The method of control variates presented in the previous subsection used the *ex-ante* formulation of the Bayesian game under consideration. We can use a player's type as a control variate, however the variance reduction achieved using this variable depends on how linear the relationship between a player's type and payoffs is. More typically, we use control variates that summarize type information. If, for instance, the type space is small, then we can apply the method of control variates separately for each type and then estimate the payoffs using the weighted average of these estimates, weighting by the probability of the type. This allows us to model nonlinear relationships between types and payoffs.

Advantage-sum method

The *advantage-sum* technique introduced by Zinkevich et al. (2006) is similar in design to the method of control variates. This technique explicitly models a sequence of events. Here, I outline the Zinkevich et al. (2006) model and review some of the existing literature that extends.

The set of all possible events is given by E and a *history* is some $h \in E^*$. We have the relation $O \subseteq H \subseteq E^*$, where H is the set of reachable histories and O is the set of terminal histories. A utility function $u : O \to \mathbb{R}$. A distribution $\sigma : H \setminus O \to \Delta(E)$ over the next event in the sequence. It is important to note that, as stated, σ describes all of the randomness in the system, that is, the strategy of every player (including Nature). There is a set $K \subseteq H \setminus O$, such that there is a known distribution $k : K \to \Delta(E)$. Let $\mathcal{K}$ be the set of all k-distributions, and Σ be the set of all σ-distributions. The distributions $k \in \mathcal{K}$ and $\sigma \in \Sigma$ *agree* if for all $h \in K$, $\sigma(h) = k(j)$. Let Σ_k be the set of all σ that agree with k.

The output from simulation, Π, is a random variable where $\Pi = u(h)$ and $h \in O$ under σ. As above, we seek an unbiased estimator of $\mathbb{E}_\sigma[\Pi]$ with minimal variance. Let $V : H \to \mathbb{R}$ be a *value function*. Given a belief $\rho \in \Sigma$, let $V^\rho(h) = \mathbb{E}_\rho[\Pi|h]$. For some $k \in \mathcal{K}$, let $Q_{V,k} : K \to \mathbb{R}$ be

$$Q_{V,k}(h) = \sum_{e \in E} V(h.e)k(e|h),$$

where $h.e$ is the sequence with e appended to h, and $k(e|h)$ is the conditional probability that e is the next event in the sequence given h under k. Define the *advantage-sum* $\hat{u}_{V,k} : O \to \mathbb{R}$ to be:

$$\hat{u}_{V,k}(h) = u(h) + \sum_{t \text{ s.t. } h(t) \in K} [Q_{V,k}(h(t)) - V(h(t+1))].$$

Let $\hat{\Pi}_{V,k} = \hat{u}_{V,k}(h)$ and $h \in O$. Zinkevich et al. prove two main theorems.

Theorem 2.1.1 (Zinkevich et al. (2006))**.** *For any $V : H \to \mathbb{R}$ and $k \in K$, $\hat{\Pi}_{V,k}$ is an unbiased estimator for* $\mathbb{E}_\sigma[\Pi]$ *for any $\sigma \in \Sigma_k$.*

Theorem 2.1.2 (Zinkevich et al. (2006))**.** *For any $k \in K$ and any $\sigma \in \Sigma_k$, $\hat{\Pi}_{V^\sigma,k}$ is a minimum variance unbiased estimator for* $\mathbb{E}_\sigma[\Pi]$.

The first theorem establishes that $\hat{\Pi}_{V,k}$, for any V, is an unbiased estimator. Therefore we restrict our attention to value functions defined by belief ρ. The second theorem establishes that if we use a value function defined by belief ρ that is correct ($\rho = \sigma$), then $\hat{\Pi}_{V^\sigma,k}$ is a minimum variance estimator.

Let $\sigma, \sigma' \in \Sigma_k$ such that if $\sigma(h) > 0$ then $\sigma'(h) > 0$ for every $h \in O$. It follows that

$$\begin{aligned} \mathbb{E}_{\sigma'}[\Pi] &= \mathbb{E}_{\sigma'}[u(h)] \\ &= \mathbb{E}_{\sigma}[u(h)\tfrac{\sigma(h)}{\sigma'(h)}]. \end{aligned}$$

This means that we can use the payoff estimate given by a simulation trace under σ to estimate the payoff under σ' for any distribution σ' meeting the criterion. In order words, we can estimate the payoff under σ' using only the data generated under σ. This is the basis for the results given by Bowling et al. (2008).

2.2 Experiments

Each of the market scenarios in the Trading Agent Competition have some stochastic component defined as part of their respective simulation. Uncertainty in the simulated environments reflects actual uncertainty agents face in these domains. In the TAC/SCM scenario, the major source of uncertainly originates from the stochastic processes that drive customer demand and component supply. In this section, I apply the method of control variates to reduce the variance in the estimates of the mean payoffs in TAC/SCM simulations.

Wellman et al. (2005a) proposed a set of control variables to derive a payoff measure called *demand-adjusted profit* (DAP). Each DAP represents an unbiased estimate of the mean payoff to a given player in a given profile. This adjustment considers the average level of demand (measured in total number of PCs requested) for each of the PC market segments: *low*, *mid*, and *high*. I collected the demand and score data from games played in the semifinal and final rounds of the TAC/SCM 2005–2009 tournaments. Table 2.1 reports the mean demand in each segment, while Table 2.2 lists the number of observations from each dataset.

Segment	Mean
Low	127,338
Mid	157,547
High	131,725

Table 2.1: Mean customer demand for each market segment in TAC/SCM.

Year	Semifinals	Finals
2005	24	10
2006	30	16
2007	15	16
2008	17	18
2009	19	16

Table 2.2: Number of observations for TAC/SCM tournament data.

Using the luck-only method, I compute DAP coefficients for each tournament year. I regress against the semifinals data and measure the variance reduction on the finals data. Because the profiles are fixed in each semifinal heat, controlling for a given strategy in the semifinals is equivalent to controlling for a given profile. For each tournament year, I measure the variance reduction with and without controlling for each player's strategy. Table 2.3 shows the results of the analysis without controlling for the strategies. The center column of the table (DAP coefficient) presents the result of a linear regression of mean agent score on demand in the respective segments. We obtain the DAP for agent i in game

x by subtracting from its actual profit an adjustment based on the demand in that game.

$$\mathrm{DAP}_i(x) = \mathrm{Profit}_i(x) - \sum_{\mathrm{seg} \in \{\mathrm{low,mid,high}\}} \delta_{\mathrm{seg}}(Q_{\mathrm{seg}}(x) - \bar{Q}_{\mathrm{seg}}), \tag{2.3}$$

where $Q_{\mathrm{seg}}(x)$ denotes the actual demand for the specified segment in game x, and $\bar{Q}_{\mathrm{seg}}$ the mean demand as presented in Table 2.1. With the exception of the 2009 tournament data, the difference in variance reduction with and without controlling for strategies is statistically insignificant, therefore I do not list the coefficients controlled for strategies in the table. In the 2009 dataset, the variance reduction obtained without controlling for strategies exceeded the corresponding method controlling for strategies by approximately 10%.

Server Version	Segment	DAP Coeff. (δ_{seg})	Low (2.5%)	High. (97.5%)	Variance Reduction
2005	Low	74.2	36.2	112.3	
	Mid	85.7	61.1	110.3	37.9%
	High	78.1	40.6	115.6	
2006	Low	61.7	42.4	81.1	
	Mid	75.9	62.7	89.1	45.8%
	High	99.2	84.5	113.9	
2007	Low	90.4	52.1	128.7	
	Mid	90.5	52.2	128.6	42.6%
	High	116.6	73.3	160.0	
2008	Low	119.1	-4.1	242.2	
	Mid	101.2	-23.9	226.2	41.4%
	High	125.3	6.0	244.7	
2009	Low	64.7	-1.3	130.7	
	Mid	85.5	41.8	129.3	37.4%
	High	35.8	-27.0	98.5	

Table 2.3: TAC/SCM tournament variance reduction using DAP, including 95% confidence intervals around the DAP coefficients.

2.3 Discussion

In this chapter, I review variance reduction techniques and describe how these approaches can be extended to a multiagent setting. I propose three classes of control variate methods for reducing variance in multiagent simulations: luck-only, strategy level, and profile level. Using the luck-only method, I demonstrate a variance reduction of at least 35% over five years of TAC/SCM tournaments. In Chapter 9, I apply these methods to reduce the variance of the payoffs in TAC/AA.

A majority of the techniques estimate parameters for empirical games in strategic form, which is the game form used throughout this thesis. Methods for estimating parameters in extensive form exist as well, with noteworthy contributions originating from the computer poker research community (Zinkevich et al., 2006; Bowling et al., 2008). The number of information states in a poker game is large, however it is dwarfed by a scenario like TAC/SCM. Thus, developing methods that can partially exploit the extensive form while maintaining feasibility is a fruitful area for future research.

Chapter 3
Evaluating Strategies

Given a set of possible strategies, what is the best strategy for a given player? How much better is one strategy compared to another? In a single-agent setting with some measure of agent utility, we can answer these questions in a straightforward manner by evaluating the utility of each strategy. The *best* strategies are the strategies with maximal utility. The degree to which one strategy is better than another is the difference in utility the agent receives between the two strategies. In a multiagent setting, evaluating a set of strategies is not as straightforward and has generated an abundance of evaluation techniques within the research community.

Solving a game in a multiagent setting is analogous to finding the optimal strategy in a single-agent setting. In game theory, the principal solution concept is the *Nash equilibrium* and the analog of a *best strategy* is the set that supports a Nash equilibrium. However, what about two strategies that do not support an equilibrium? Can one be *better* than the other? In this chapter, I propose a principled strategy-evaluation measure and contrast this with other existing methodologies. The proposed evaluation measure is termed *NE regret* and corresponds to the strategy-regret of a player when the opponent profile, called a *background context*, in the strategy-regret calculation is a Nash Equilibrium. Using *NE regret* as a comparison criterion induces a *rationally consistent ranking* over strategies, termed an *NE-response ranking*.

Section 3.1 introduces NE-regret, while the discussion of NE-response ranking is deferred until Section 3.3. Preceding the description of NE-response rankings, I illustrate a visualization technique for compactly displaying the stability of profiles in pure strategy space. Deviation graphs and the contoured extension given in Section 3.2 were observed by Wellman et al. (2005a) and Jordan et al. (2007) to be a useful tool for summarizing a strategy space. I illustrate the usefulness of the evaluation measure and ranking technique through an analysis of the Annual Computer Poker Competition in Section 3.4. The resulting NE-response ranking differs from the rankings given by the two methods used in the competition.

3.1 Equilibrium-Based Strategy Regret

In a single-agent setting, we can compare strategies by the difference in their respective utilities. In a multiagent setting the corresponding utilities depend on the opponent profile. If the differences in utilities are invariant over all opponent profiles, then we can construct a measure similar to the one used in the single-agent setting. However, this condition will not be met in general.

Strategy regret (Definition 8) provides a method by which we can evaluate each strategy in a set of candidate strategies. This measure is dependent on the choice of opponent profile, forcing us to characterize what is reasonable for an opponent profile. The evaluation measure I propose in this chapter restricts the set of allowable opponent profiles to those supporting Nash equilibria. There may be multiple equilibria, in which case there may be multiple regret values for a given strategy.

Definition 13 (NE Regret). *The* NE regret *of strategy $s_i \in S_i$ for player i is $\delta_i(s_i|\sigma_{-i}^{NE})$ where σ^{NE} is a Nash equilibrium of game Γ.*

In a symmetric game with symmetric Nash equilibrium σ^{NE}, all strategies in support of σ^{NE} have an NE regret of zero. The use of equilibrium profiles as a basis for opponent profiles is founded on two lines of reasoning. First, Jordan et al. (2007) make the assumption that, all else being equal, stable profiles are more likely to be played in practice. The assumption is echoed in this work and is empirically supported by Erev and Roth (1998). Second, stable background contexts are intuitively appealing because they are equilibrium points for rational agent play. Using NE regret as a strategy selection criterion can differ significantly from other criteria such as *minimax-regret*.

Aside from measuring stability, strategy regret can reveal additional useful information about a strategy. For instance, defining a feature-based or parameterized strategy is typical for complex scenarios, such as TAC/SCM, where the agent is making many different complex decisions. If a strategy is decomposable into a set of behavioral features, this measure can ascribe a value or utility to each of its constituents. From a designer's perspective, this is particularly useful in heuristically constructing novel strategies from the evaluated features of similar strategies.

3.2 Contour Deviation Graphs

A *contour deviation graph* is a visualization technique that describes the quality of strategies through the relative stability of the profiles they support. I illustrate this technique through an analysis of the simple game given in Table 3.1. The game has two players and three actions. The profile CC is the sole pure strategy Nash equilibrium of the game.

Row \ Col	A	B	C
A	(4,4)	(5,1)	(3,5)
B	(1,5)	(2,2)	(1,3)
C	(5,3)	(3,1)	(5,5)

Table 3.1: Two-player, three-action game example game.

However, simply identifying the equilibria of a game does not give us the whole picture. For instance, consider Figure 3.1. The left half of the figure shows the connectedness of the profile space, where a node in the graph is a profile and an edge represents a possible deviation from the deviation set (Definition 3).

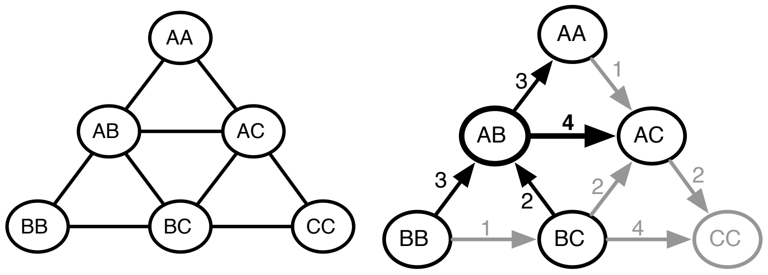


Figure 3.1: Deviation space (left) and deviation gain (right) of the two-player, three-action game in Table 3.1, with an emphasis on the AB profile.

The right half of the figure has more information overlaid on the edges. The edge weights represent the regret of the player playing its strategy in the source profile when given the option of switching to the corresponding strategy in the destination profile. For instance, consider the profile AB. When analyzing AB, only the deviating profiles AA, BB, BC, and AC affect the stability of AB. The profiles not in AB's deviation set have been colored gray as well as the deviations edges not used in its computation. From this we can see that the deviation from AB giving the largest gain is AC.

Looking at the deviation gain graph as a whole, we can also make observations about individual strategies. The strategy C seems particularly strong in this setting. It supports the pure strategy Nash equilibrium and, additionally, is an improving deviation in five out of six possible deviations (edges in the graph where a player changes its strategy to C). Similarly, A is improving in four out of six possible deviations. Finally, we can see that B is a particularly poor choice in that there are no improving deviations using B.

Figure 3.2 gives an alternative view of the deviation space using contour levels to indicate the stability of a profile. Edges in the graph represent the best deviation from the profile. The magnitude of the gain from deviation is represented by the contour level. In the contour graphs, modelers make qualitative judgements of strategy robustness by considering the relative contour levels in which the strategies appear as well as the frequency and magnitude of the deviation gains attributed to the strategy. For instance, the Nash equi-

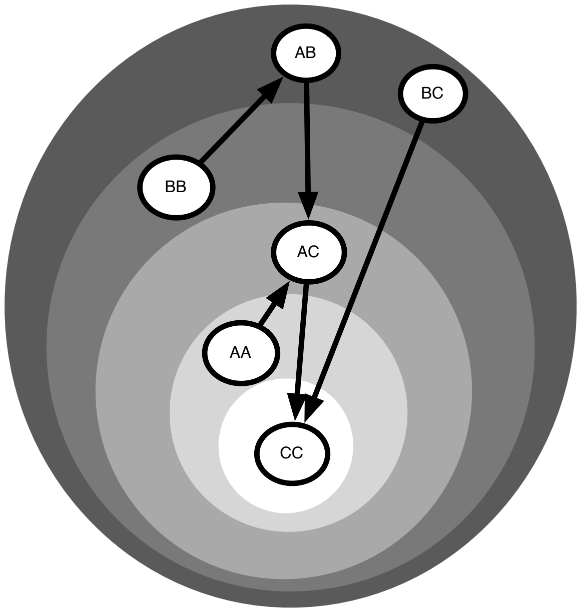


Figure 3.2: Contoured deviation graph of the three action game in Table 3.1.

librium, CC, is the most stable profile with a gain from deviation of zero. It is placed in the innermost contour. Proceeding outward, each contour level contains progressively less stable profiles, increasing in one unit of regret for each contour level. Thus, AA has a regret of one and AB has a regret of four. Using this visual representation, strategy B is quickly identified as a poor strategic choice, given that the profiles it supports occupy the outermost contour levels. Conversely, C is identified as a strategically stable choice.

3.3 NE-Response Ranking

The NE-regret measure of Section 3.1 assigns a level of regret to each strategy and thus induces a ranking over strategies. The notion of comparing or ranking strategies is not novel. For instance, in the annual TAC tournament[1], participants are ranked according to their relative scores in a *multi-stage tournament*. The TAC tournament generates results

[1]A TAC tournament is typical of many competitive tournament settings and is similar in structure to the NCAA Basketball Championship structure. There is an initial seeding round (round-robin style tournament) in which teams determine their initial bracket. Each subsequent round eliminates half of the tournament participants.

from one particular path of combinations, and success in the tournament is rightly regarded as evidence for agent quality.

Consider the design of a multi-stage tournament used to compare strategies from the game in Table 3.1. The scenario is a two-player, symmetric game with three strategies. During a round in a tournament, two strategies are compared. The winner is the player that has a higher score in the profile. For instance, when the strategies A and C are compared, we see that C is preferred to A. This is because, in the profile AC, the agent playing C receives a payoff of 5 and the agent playing A receives a payoff of 3. A tournament is formed by comparing two strategies, determining a winner, and then comparing the surviving strategy to the third remaining strategy. Given the payoffs from Table 3.1, there are three possible outcomes of the tournament shown in Figure 3.3.

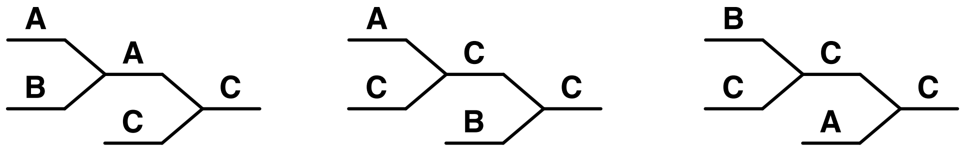


Figure 3.3: Possible multi-stage tournament outcomes for the game in Table 3.1.

First, notice that, in all cases, C wins the tournament. This supports the qualitative assessment of strategies in Section 3.2. However, note that the tournament does not produce a complete ranking, since the first losing strategy is never compared to the third strategy. Moreover, strategies will have no opportunity for self-play, thus omitting valid strategic aspects of the scenario in the comparison. Finally, in a multi-stage tournament the outcome of the tournament is dependent on the ordering. Consider a similar design for a rock-paper-scissors tournament. In Figure 3.3 each of A, B, C strategies would be replaced by *rock*, *paper*, and *scissors*, respectively. Given the three initial tournament configurations, each strategy wins the tournament exactly once.

Alternatively, we could construct a *round-robin* tournament that includes self-play and compare the average score of each strategy, as shown in Table 3.2. The preferred ranking

Strategy	Average Score
A	4.00
B	1.75
C	4.50

Table 3.2: Average score for each strategy in a round-robin tournament of the game in Table 3.1.

given by the round-robin tournament is $C \succ A \succ B$, which is consistent with the qualitative ranking given in Section 3.2. However, round-robin tournaments fail to provide a reasonable ranking in cases like prisoner's dilemma, where the ranking would consider a dominated strategy as preferable to the dominant strategy.

We would like the ranking to be consistent with the preferences of rational players in the game. The proposed *NE-response ranking* is grounded in these preferences. Before defining the NE-response ranking, I discuss a set of properties a ranking over strategies should have that are similar in spirit to the Bernheim (1984) and Pearce (1984) concept of *rationalizable strategies*.

Definition 14 (Rationalizable Strategies (Bernheim, 1984; Pearce, 1984))**.** *Let* $\Gamma = \langle N, S, u\rangle$*,* $R^{(0)} = S$*, and* $R^{(i)} = \mathfrak{B}(R^{(i-1)})$*. The* rationalizable strategies *are given by* $R = R^{(\infty)}$.

The first property is termed *rational support*. This property states that there is some mixed opponent profile for which the ranking respects a rational player's preference relation with respect to that opponent profile.

Definition 15 (Rational support)**.** *For a game* Γ *and player* i *with strategy set* S_i*, a binary relation* $\succeq_i$ *on* S_i *has* rational support *if*

- $\succeq_i$ *is a total preorder on* S_i
- $\exists \sigma_{-i} \in \Delta\left(S_{-i}\right)$ *such that for every pair of strategies* $s_i, \hat{s}_i \in S_i$,

$$s_i \succeq_i \hat{s}_i \implies u_i(s_i, \sigma_{-i}) \geq u_i(\hat{s}_i, \sigma_{-i}).$$

I propose a family of ranking methods, called σ-response rankings. These methods use a mixed profile σ as a *background context.* For each strategy s_i, we compute the payoff player i receives by playing s_i when the other players play σ_{-i}.

Definition 16 (σ-response ranking)**.** *For a game Γ, player i, and mixed profile $\sigma_{-i} \in \Delta(S_{-i})$, a σ_{-i}-*response ranking *denoted by $\succeq^{\sigma_{-i}}$ is a ranking over player i's strategies that is rationally supported by σ_{-i}. A σ-response ranking is a joint ranking $\succeq^{\sigma} = (\succeq^{\sigma_{-1}}, \ldots, \succeq^{\sigma_{-n}})$ where each component is a response ranking for a respective player.*

Note that σ_{-i}-response rankings are defined with respect to a specific player i in Γ, however if Γ is symmetric, then the ranking is the same for each player.

Any σ-response ranking has rational support given by σ, however not all σ are justifiable as background contexts under rational behavior. From Bernheim (1984) and Pearce (1984), we can prune strategies that are not *rationalizable* and reduce the set of justifiable background contexts to those that are mixtures over rationalizable strategies. Let *rationalizable rankings* be rankings that have rationalizable support. A joint ranking $\succeq$ is a *rationalizable ranking* if there is a rationalizable mixed profile σ such that $\succeq_i$ is supported by σ_{-i} for each player i.

We could identify the set of rationalizable rankings and disregard those that are not rationalizable. However, this raises the existential question: *are there any joint rationalizable rankings that are not justified by rational behavior?* Consider the matching pennies game given in Table 3.3. Each player's full strategy set is rationalizable. Therefore, $(\{H \succ_R T\}, \{h \succ_C t\})$ is a jointly rationalizable ranking supported by the rationalizable profile Th. In this case, the row player is playing a strategy it considers inferior according to its stated ranking. Moreover, there is no profile that supports this joint ranking without mixing over inferior strategies for the column player. Therefore $(\{H \succ_R T\}, \{h \succ_C t\})$ is not consistent with rational strategic behavior.

Now that we know some rationalizable rankings are not justified, the question remains as to whether any joint ranking $\succeq = (\succeq_1, \ldots, \succeq_n)$ is justifiable under rational strategic

Row \ Col	h	t
H	(1,-1)	(-1,1)
T	(-1,1)	(1,-1)

Table 3.3: Matching pennies game.

behavior. To justify a joint ranking with rational behavior, I propose a property called *rational consistency* that is defined subsequently. Plainly speaking, rational consistency requires a profile that supports a joint ranking mix over strategies that are consistent with the individual players' rankings.

Definition 17 (Rational consistency)**.** *For a game* Γ *and joint ranking* $\succeq = (\succeq_1, \ldots, \succeq_n)$*,* $\succeq$ *satisfies* rational consistency *if there exists some profile* $\sigma \in \Delta^\Gamma$ *such that for every player* $i \in N$*,*

- $\succeq_i$ *is rationally supported by* σ_{-i}*;*
- $\forall s_i \in S_i$ *if* $\sigma_i(s_i) > 0$ *then* s_i *is maximal under* $\succeq_i$*.*

If σ supports a rationally consistent $\succeq$, the two conditions imply that σ is a Nash equilibrium. Therefore the set of rationally consistent joint rankings is given by the set of σ-response rankings, where σ is a Nash equilibrium. I term an element of this set an *NE-response ranking*.

Definition 18 (NE-response ranking)**.** *For a game* Γ*, an* NE-response ranking *is a* σ*-response ranking where* σ *is a Nash equilibrium in* Γ*.*

Consider, again, the game presented in Table 3.1. There is a degenerate mixed-strategy equilibrium where each player plays C with probability 1. If we allow a single player deviation to each of A, B, and C, we can calculate our σ-response ranking. Table 3.4 illustrates deviation gain for each strategy from the Nash equilibrium background context as well as two other background contexts where players play A and B with probability 1, respectively. Using A as a background context, the resultant ranking is $C \succ A \succ B$, whereas B gives the ranking $A \succ C \succ B$. The background context of B does not seem to capture what we observed by visually analyzing the deviations. When using the Nash equilibrium context

(C), we do see a ranking $C \succ A \succ B$ that is consistent with our prior qualitative observations. Notice that both background contexts A and C induce the same rationally-consistent ranking, even though A is not a Nash equilibrium. In this case, the profile of 100% A is an approximate Nash equilibrium with a regret of 1.

Background Context	A	B	C
100% A	0	-3	1
100% B	3	0	1
100% C	**-2**	**-4**	**0**

Table 3.4: Pure-strategy deviation gain for three background contexts. 100% C is the Nash equilibrium background context and is in **bold** font.

3.4 Computer Poker Competition

The Computer Poker Competition[2] is an annual competition that evaluates research in autonomous computer poker agents. The competition has existed since 2006 with over 40 entrants competing. The 2009 tournament[3] consisted of two 2-player (heads-up) competitions and one 3-player competition. The 2-player competitions include a no-limit and limit variant of Texas Hold'em poker. In total, 13 bots participated in the limit competition and 5 in the no-limit. As part of the tournament, an empirical payoff matrix is created that contains mean payoff estimates for each profile. For the limit competition, each bot played 75 matches of 3000 hands, while in the no-limit competition each bot played 60 matches of 3000 hands.

Figure 3.4 shows the contour deviation graph for the 2009 no-limit competition bots. Each step subsequent level indicates a unit increase in regret. From the graph, we observe that Tartanian3 and Tartanian3RM are not strategically stable strategies. They only support profiles that occupy the outer levels of the contour deviation graph. However, BluffBot4,

[2]See `http://www.cs.ualberta.ca/~pokert/`

[3]See `http://www.cs.ualberta.ca/~pokert/2009/results/`

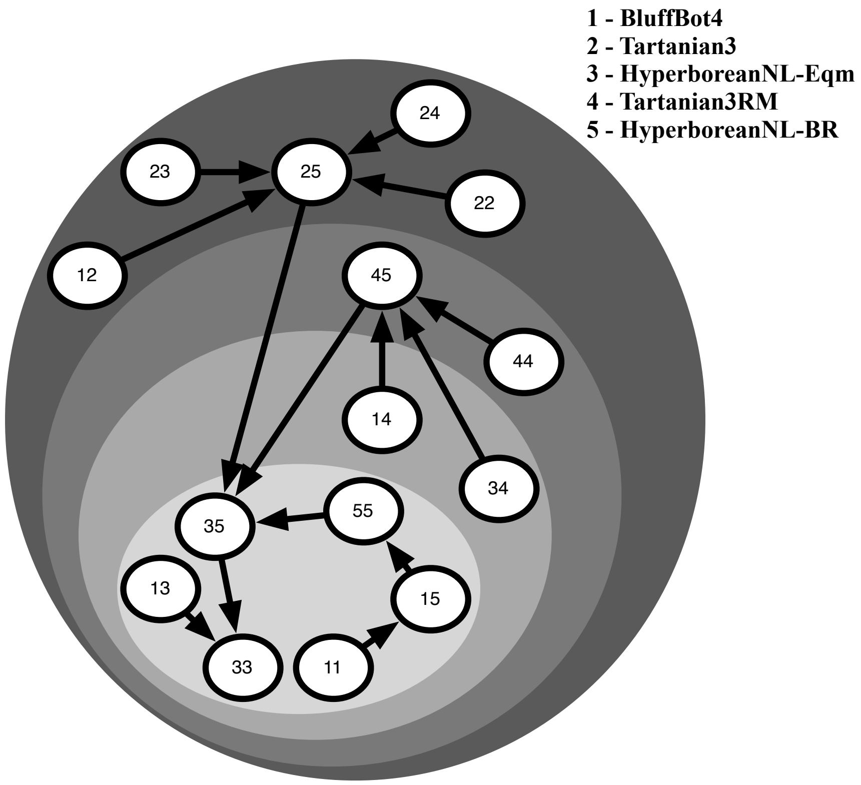


Figure 3.4: The 2009 no-limit competition bots.

HyperboreanNL-Eqm, and HyperboreanNL-BR all support profiles that are in the innermost level. The profile consisting of only HyperboreanNL-Eqm is a pure-strategy Nash equilibrium.

Figure 3.5 shows the deviation graph for the 2009 limit competition bots. The color of the circle implies its stability, with black being the least stable and white being the most stable. The darker, distal profiles all contain GS5Dynamic or tommybot as selected strategies. In the upper right of the figure, we see stable profiles that contain Rockhopper. The white, proximal profile contains all GGValuta, which is a pure-strategy Nash equilibrium.

For each competition, bots are ranked according to two methods: *bankroll* and *run-off*. Bots in the bankroll competition are evaluated by their aggregate number of chips won over all hands. Because each bot plays each other bot an equal number of times, this is equivalent to ranking by a bot's mean score against an opponent playing a uniform distribution over all bots. In other words, a *uniform-response ranking*.

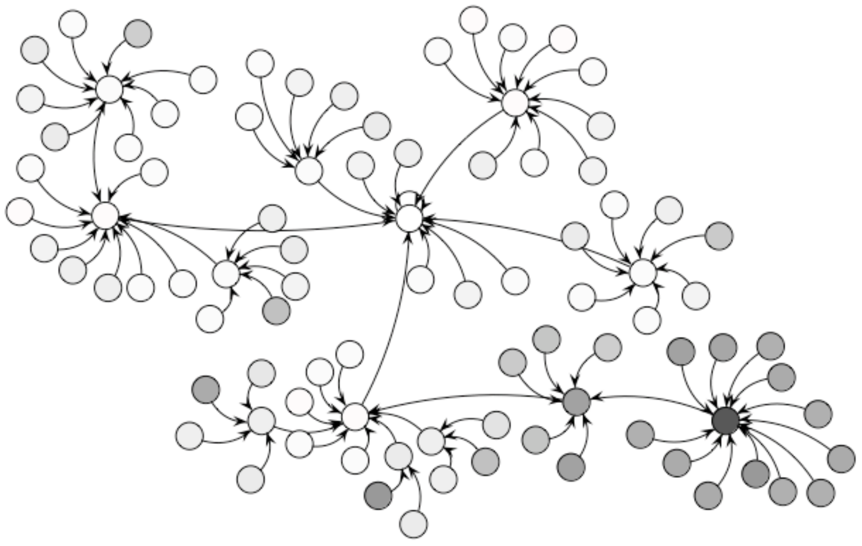


Figure 3.5: The 2009 limit competition bots.

The run-off method[4] recursively eliminates bots from consideration. A bot's rank is determined by how many rounds it survives. Starting with a set S of all bots, compute the *uniform-response ranking* with respect to S. Eliminate the worst-ranked bot and recurse. The run-off method bears a strong resemblance to the *iterated-regret-minimization* solution concept introduced by Halpern and Pass (2009). The tournament designers describe the run-off method as an "equilibrium competition", with the idea that first place bot should be able to beat the second placed bot in a heads up match and that, in general, the i^{th}-place bot should have a higher bankroll than j^{th}-place bot when considering only matches between the top j bots for $i < j$.

Table 3.5 highlights the differences in rankings between the various methods for the 2009 no-limit bots. Not surprisingly, the bankroll ranking differs from the run-off and NE-response rankings (shown in bold). Both the run-off and NE-response method reverse the order of the top two bots, but NE-response also reverses the ranking of the bottom two bots. If we assume that players are rational in selecting which bot to play, bankroll ranking does not yield reasonable results. For instance, HyperboreanNL-Eqm is the sole

[4]See `http://www.cs.ualberta.ca/~pokert/2009/rules.php`

survivor of *iterated elimination of strictly dominated strategies* and therefore should be in the highest ranked. The large difference in bankroll between HyperboreanNL-Eqm and HyperboreanNL-BR results from HyperboreanNL-BR's ability to exploit the weaker bots, which rational opponents would not use.

Bot	Bankroll	Run-off	NE-Response Ranking
HyperboreanNL-BR	1	**2**	**2**
HyperboreanNL-Eqm	2	**1**	**1**
BluffBot4	3	3	3
Tartanian3RM	4	4	**5**
Tartanian3	5	5	**4**

Table 3.5: Comparison of ranking methods for 2009 no-limit bots.

Table 3.6 displays the corresponding differences in rankings between the various methods for the 2009 limit bots. Here again, the bankroll ranking differs substantively from the other rankings. In particular, HyperboreanLimit-BR is ranked second in the bankroll ranking and fifth in the others, largely due again to its ability to exploit weaker bots.

Bot	Bankroll	Run-off	NE-Response Ranking
MANZANA	1	**3**	**2**
HyperboreanLimit-BR	2	**5**	**5**
GGValuta	3	**1**	**1**
HyperboreanLimit-Eqm	4	**2**	**3**
Rockhopper	5	**4**	**4**
Sartre	6	**7**	**7**
Slumbot	7	**6**	**6**
GS5	8	8	8
AoBot	9	9	9
LIDIA	10	**11**	**11**
dcurbhu	11	**12**	**12**
GS5Dynamic	12	**10**	**10**
tommybot	13	13	13

Table 3.6: Comparison of ranking methods for 2009 limit bots.

3.5 Discussion

The NE-regret and NE-response rankings I discuss in this chapter provide a principled methodology to evaluate strategies. Through a case study of the Annual Computer Poker Competition, I observe differences in ranking between the regret-based method and the existing techniques, while highlighting inconsistencies in the latter from a rational-behavior perspective.

Using competitions to motivate research in complex environments is an increasingly common research technique within the artificial intelligence and robotics communities (Gini, 2009). In a multiagent setting, care must be taken by the designers to align the incentives of the entrants with the research goals of the competition. The Trading Agent Competition, which promotes research into the problems faced by designers of trading agents and the markets in which they participate, is an excellent example. Each year a tournament is run to determine the best set of agents for a specific market scenario. The hope is that entrants are creating competent agents that tell us something about how to behave optimally in these various markets. However, from a entrant's perspective, there is a strong incentive to design an agent to *win* the tournament. An agent that wins the tournament may act pathologically, with respect to the underlying research goals, in order to do so. For instance, an agent may be willing to lose a large amount of money as long the agent makes more money than its opponents. While this behavior is reasonable with respect to the entrant's goals, it is not reasonable with respect to the goals of an agent in the actual market scenario. Using a multi-stage or round-robin tournament, like the bankroll rankings of the Computer Poker Competition, does not always correctly align incentives of the entrants. In general, this is because an equilibrium strategy in the competition may not be an equilibrium strategy in the stage game, the game on which research is focused. However, the NE-response ranking method of this chapter does correctly align the incentives of the competition entrants and the research goals.

Chapter 4

Rational Closure in Empirical Games

Many computational problems in game theory, such as finding Nash equilibria, are hard to solve. This limitation forces analysts to limit attention to restricted subsets of the entire strategy space. In this chapter, I develop algorithms to identify relevant subsets of the strategy space under given size constraints. First, I modify an existing family of algorithms for two-player games to compute rationally-closed strategy sets for n-player games. I then extend these algorithms to apply in cases where the utility function is partially specified, or there is a bound on the size of the restricted profile space. Finally, I evaluate these algorithms on various game classes and compare their performance to that of other known algorithms.

4.1 Restricted Games

Researchers in multiagent systems often appeal to the framework of game theory. It has well understood solution concepts and a general formalism that allows analysts to model a wide variety of scenarios. However, many game-theoretic analysis procedures—such as solving a game—are computationally complex. All else being equal, smaller games are easier to analyze. This is exemplified in computational game-theoretic literature, where the complexity of some of the most basic operations are PPAD-complete or NP-complete. Therefore, methods that reduce the size of a game can have a large impact on the feasibility of analysis. For example, decreasing the effective number of players by hierarchical reduction (Wellman et al., 2005b) or clustering (Ficici et al., 2008) can reduce the number of profiles exponentially. We can also reduce the size of a game by pruning strategies from some player's strategy set. If strategies that are not relevant to the analysis can be identified, then they can be safely pruned. Determining if a strategy is irrelevant can be computationally complex, depending on the type of analysis undertaken.

Definition 19 (Restricted Game)**.** *Let* $\Gamma_{S\downarrow X}$ *be a* restricted game *with respect to the* base game Γ*, where each player* i *in* $\Gamma_{S\downarrow X}$ *is restricted to playing strategies in* $X_i \subseteq S_i$*.*

Our goal is to find a restricted game that is minimal in size, yet conveys approximately the same relevant information as the base game. Obviously, what information is relevant depends on the exact type of strategic analysis, however one common vein in empirical game-theoretic analysis is computing the regret of profiles. The regret computation is a core component of the analysis of Chapter 3. In this context, we desire a restricted game such that the regret of any profile with respect to the restricted game is approximately the same as the regret of that profile with respect to the base game. Of course, when we prune strategies, we lose the ability to calculate the regret of profiles not within the restricted game. However, pruning irrelevant strategies can dramatically reduce the size of the game. As analysts, we can determine an acceptable tradeoff between this loss and the set of methods made feasible to us by the reduction.

From a player's perspective, the regret of playing a particular strategy in a profile depends on the other available strategies in the game. For a given profile, only the strategies that are improving deviations give the player positive regret and thus are relevant to the regret calculation. Suppose we can identify a joint strategy set such that, for each player and each available profile in the restricted space, the player's improving deviations are contained in the restricted set of strategies for that player. Strategies outside that set are irrelevant from the point of the regret analysis. The restricted game formed from the joint strategy sets is *closed*, in the sense that a rational player would not choose to play strategies outside those of the restricted game, given knowledge that the other players choose strategies within their restricted strategy sets. Therefore, we can prune these irrelevant strategies and reduce the size of the game.

4.2 Pruning Strategies Through Strategy Elimination

In this section, I review some of the available algorithms used to prune strategy sets. *Iterated elimination of strictly dominated strategies* (IESDS) is an obvious first choice. IESDS removes dominated strategies from the players' strategy sets. Dominated strategies are *non-rationalizable* (Bernheim, 1984; Pearce, 1984). In two-player games, only *rationalizable strategies* remain after IESDS, but this is not necessarily the case for more than two players.

Another possibility is to compute all of the Nash equilibria of the game and keep only the strategies constituting those equilibria. This approach has some obvious downsides. Foremost, if identifying equilibria is the goal, then computing the equilibria as a preprocessing step does not reduce the complexity of the analysis as a whole. Moreover, computing equilibria is hard. For instance, computing a single NE is hard (PPAD-complete, see Chen and Deng (2006) for two-player games; Daskalakis and Papadimitriou (2005) for three-player games; and Daskalakis et al. (2005) for four or more players), let alone all NE.

Conitzer and Sandholm (2005) introduce a general *eliminability criterion* for two-player games. The authors compute whether a strategy is eliminable by solving a mixed integer program, which implicitly considers the rationalizable and NE solution concepts discussed so far. However, whereas iterated elimination using this criterion may not rule out equilibrium strategies in the base game, it may introduce NE in the restricted game that are not equilibria in the base game. This is undesirable; we would like to know that profiles identified as NE in the restricted game persist as NE in the base game. Persistence of this type leads to the idea of *rational closure*.

4.3 Rational Closure

In the previous section, I informally discussed properties of rationally closed sets. A restricted strategy set, X, is *rationally closed* if the set is closed under a given best-response correspondence. In this section, I review two formal definitions of rational closure introduced by Basu and Weibull (1991) and Harsanyi and Selten (1988), respectively. The difference between these two concepts is subtle and lies in the use of two different best-response correspondences, $\mathfrak{B}(X)$ and $\mathfrak{B}^{\dagger}(X)$, respectively. Strategies in $\mathfrak{B}^{\dagger}(X)$ are best-responses to correlated opponent mixtures, whereas strategies in $\mathfrak{B}(X)$ are best-responses to independent mixtures.

Definition 20 (Closed under rational behavior (Basu and Weibull, 1991))**.** *A set of profiles $X \subseteq S$ is*
- *closed under rational behavior (CURB) if $\mathfrak{B}(X) \subseteq X$.*
- *a minimal CURB set if no proper subset of X is CURB.*

Definition 21 (Formation (Harsanyi and Selten, 1988))**.** *A set of profiles $X \subseteq S$ is*
- *a formation if $\mathfrak{B}^{\dagger}(X) \subseteq X$.*
- *a primitive formation if no proper subset of X is a formation.*

Harsanyi and Selten (1988) use the term primitive formation as the analog of a minimal CURB set in the CURB framework. I use the term *minimal formation* interchangeably with primitive formation as well as *min-CURB* for minimal CURB set.

Each formation contains at least one Nash equilibrium in the base game. Moreover, the regret of a profile in a formation, with respect to the restricted game defined by the formation, is equal to the regret of the profile with respect to the base game.

Theorem 4.3.1. *If X is a formation and $\sigma \in \Delta^X$, then $\epsilon(\sigma|\Gamma) = \epsilon(\sigma|\Gamma_{S\downarrow X})$.*

Proof. By construction, for each player $i \in N$, $S_i \supseteq X_i \supseteq \mathfrak{B}_i(\sigma_{-i})$. Therefore,

$$\begin{aligned}
\epsilon(\sigma|\Gamma) &= \max_{i \in N} \max_{s_i \in S_i} u_i(s_i, \sigma_{-i}) - u_i(\sigma) \\
&= \max_{i \in N} \max_{s_i \in \mathfrak{B}_i(\sigma_{-i})} u_i(s_i, \sigma_{-i}) - u_i(\sigma) \\
&= \max_{i \in N} \max_{s_i \in X_i} u_i(s_i, \sigma_{-i}) - u_i(\sigma) \\
&= \epsilon(\sigma|\Gamma_{S \downarrow X}).
\end{aligned}$$

□

Rational closure is central to many set-valued solution concepts. A *set-valued* solution concept assigns to each game Γ a collection of product sets of strategies. Each element $X \in \varphi(\Gamma)$ is defined as $X = \times_{i \in N^\Gamma} X_{-i}$ where $X_i \subseteq S_i$. Set-based solution concepts using some notion of rational closure include *persistent retracts* (Kalai and Samet, 1984), *minimal prep sets* (Voorneveld, 2004, 2005), *minimal CURB sets* (Basu and Weibull, 1991), and *primitive formations* (Harsanyi and Selten, 1988).

Identifying minimal CURB sets is complex, even in the two-player case. Benisch et al. (2006) introduce a set of algorithms for finding these sets in two-player games. In the following section, I describe an extension of their algorithm for n-player games that identifies the primitive formation sets. These algorithms are closely related to the generalized-saddle-point algorithms of Brandt et al. (2009). Because $\mathfrak{B}(X) \subseteq \mathfrak{B}^\dagger(X)$, primitive formations weakly contain minimal CURB sets.

4.4 Algorithms for Finding Primitive Formations

Benisch et al. (2006) give algorithms for finding all minimal CURB sets, a sample minimal CURB set, and the smallest minimal CURB set. The three algorithms all rely on a subroutine that computes $\mathfrak{B}_i(X)$ for some $X \subseteq S$. The authors determine these strategies by solving a *feasibility* problem: checking whether each strategy is a best-response to some opponent mixture. When players mix over strategies independently, the feasibility problem is nonlinear for more than two players. Therefore, the authors restrict their attention to two-player games.

I provide an extension of the Benisch et al. algorithms that is instead based on the computation of $\mathfrak{B}_i^\dagger(X)$. Allowing correlated opponent play restores the linearity of the feasibility problem, however we forgo minimal CURB sets with primitive formations remaining in their stead. The Correlated-All-Rationalizable (CAR) algorithm gives the extension of the Benisch et al. All-Rationalizable algorithm to n-player games with correlated opponent play.

Algorithm 1 CAR(S_i, X_{-i}, u_i)

```
$S_i^* \leftarrow \emptyset$
for $s_i \in S_i$ do
    if solution to the following linear feasibility problem exists
    find $p_x$ such that
        $\sum_{x \in X_{-i}} p_x = 1$
        $p_x \geq 0 \quad (\forall x \in X_{-i})$
        $\sum_{x \in X_{-i}} p_x u_i(s_i, x) \geq \sum_{x \in X_{-i}} p_x u_i(\hat{s}_i, x) \quad (\forall \hat{s}_i \in S_i \setminus \{s_i\})$
    then
        $S_i^* \leftarrow S_i^* \cup \{s_i\}$
return $S_i^*$
```

The Minimum-Containing-Formation (MCF) algorithm is the extension of the two-player Min-Containing-CURB algorithm (Benisch et al., 2006). Like the two-player algorithm, MCF uses seed strategies to generate a minimum (primitive) formation containing the seed strategy. In many cases, the minimum formation is also the minimum CURB set containing the seed strategies.

The algorithms for finding all minimal formations, a sample minimal formation, and the smallest minimal formation are constructed by substituting the MCF algorithm for the Min-Containing-CURB algorithm in the respective procedure (Benisch et al., 2006). Note that seed strategies must be provided for $n - 1$ of the players in the non-symmetric cases and one player in the symmetric case.

Algorithm 2 MCF($s_2, \ldots, s_n, \langle N, (S_i), (u_i)\rangle$)

```
for $2 \le j \le n$ do
    $S^*_j \leftarrow \{s_j\}$
converged $\leftarrow$ false
while $\neg$converged do
    converged $\leftarrow$ true
    for $1 \le j \le n$ do
        $\hat{S}_j \leftarrow$ CAR$(S_j, \times_{k \in N \setminus \{j\}} S^*_k, u_j)$
        if $\hat{S}_j \setminus S^*_j = \emptyset$ then
            $S^*_j \leftarrow \hat{S}_j$
            converged $\leftarrow$ false
return $\langle S^*_1, \ldots, S^*_n, u\rangle$
```

4.5 Partially-Specified Games

The utility $u(s)$ is defined for every profile s in a game Γ. In empirical games, utility estimates are constructed from Θ, however there may exist a set $M \subseteq S$ such that none of the observations in Θ correspond to any $s \in M$. In this section, I explore methods of assigning utility estimates to those *missing profiles* M and the consequences of these methods on the primitive formation algorithms of Section 4.4. I consider the domain of $u(\cdot)$ to be $S \setminus M$ and, if $M \neq \emptyset$, I call $u(\cdot)$ a partially-specified utility function (PSU).

The CAR algorithm uses the payoffs given by the utility function $u(\cdot)$ to determine which strategies are best responses to mixtures over X_{-i}. The algorithm assumes that we have an estimate for the utility of each profile $s \in S_i \times X_{-i}$, however this may not be the case. First, consider the use of a *default utility* $\tilde{u}$ for $s \in M$.

If $s \in M$, we use $\tilde{u}$, in place of $u_i(s)$, when solving the feasibility problem in the CAR algorithm. This may give an unduly pessimistic view of the relative utilities associated with a strategy $s_i \in S_i$, when determining if s_i is a best response. This can occur in two ways for a given opponent profile $x_{-i} \in X_{-i}$:

- $(s_i, x_{-i}) \in M$ and the actual utility $u(s_i, x_{-i}) > \tilde{u}$,
- for some $\hat{s}_i \in S_i \setminus \{s_i\}$, $(\hat{s}_i, x_{-i}) \in M$ and the actual utility $u(\hat{s}_i, x_{-i}) < \tilde{u}$.

Given either of those scenarios, we could exclude s_i from S^*_i given our current partially specified utility function $u(\cdot)$, only to learn later that s_i is rationalizable for some mixture over X_{-i} once we have estimates for the associated utilities. Therefore, once again, we could have a profile that is an equilibrium in the restricted game, but not in the base.

We would like to eliminate as many strategies as our observations allow, but not at the expense of removing strategies in the support of a minimal formation. One effect of erroneous strategy elimination is the loss of ability to make general claims about NE using only the subset of the profile space returned by the formation-finding algorithms. Notice that we can straightforwardly check whether $X \subseteq S$ is a formation, if $\mathcal{D}(X) \cap M = \emptyset$. However, restricting formation analysis to subspaces with estimated utilities is too strong a requirement.

Instead of using a default utility, if we make *optimistic assumptions* on the utilities of missing profiles, we can still guarantee the sets returned by any of the formation finding algorithms are actually formations, albeit not necessarily primitive formations. As in the consideration of default utility values, I focus on the relative utilities of a strategy $s_i \in S_i$. Let $[u_i^-, u_i^+]$ be known bounds on the utility of player i. In the algorithm, the utilities for player i of the missing profiles are assigned to either u_i^- or u_i^+, according to the following two rules:

- if $(s_i, x_{-i}) \in M$, then u_i^+ is used;
- for $\hat{s}_i \in S_i \setminus \{s_i\}$, if $(\hat{s}_i, x_{-i}) \in M$, then u_i^- is used.

Let MCF-PSU be the extension of the MCF algorithm using the previous rules for missing profiles.

Theorem 4.5.1. *If X is the set returned by MCF and X_{PSU} is the set returned by MCF-PSU for the same parameters, then $X \subseteq X^{PSU}$.*

Proof. I use induction on the size of the missing profile set. For the **base case**, let $M = \{s\}$. Assume $X^{\text{PSU}} \subset X$. Therefore, for some player i, there is a strategy $s_i^* \in X_i$ that is not in X_i^{PSU}. Because $s_i^* \in X_i$, there is some iteration in MCF such that the LFP is satisfied for s_i^* in the CAR algorithm. Let p^* be the probability vector that solves the LFP. Three cases can occur: (i) s is a profile in the left-hand-size (LHS) of the utility-based constraints, (ii) s is a profile in the right-hand-size (RHS), (iii) s is neither a profile in the LHS nor the RHS. In case (i), $u_i(s) \leq u_i^+$ implies that the LHS constraints are weakly greater under CAR-PSU, and p^* satisfies the new LFP. In case (ii), $u_i(s) \geq u_i^-$ implies that the RHS constraints are weakly less under CAR-PSU, and p^* satisfies the new LFP. In case (iii) the CAR-PSU utilities are unchanged, so p^* trivially satisfies the LFP. Therefore, the PSU-based LPF is satisfied for s_i^* under CAR-PSU and $s_i^* \in X_i^{\text{PSU}}$ Hence, by contradiction, we have $X \subseteq X^{\text{PSU}}$.

For the inductive step, assume $X \subseteq X^{\text{PSU}}$ where M_k is the missing profile set and $|M_k| = |k|$. I use the same reasoning to conclude that under $M_k + \{s\}$, $X \subseteq X^{\text{PSU}}$ — the LHS (RHS) constraints weakly increase (decrease) when under $M_k + \{s\}$. Therefore, by induction, $X \subseteq X^{\text{PSU}}$ under any countable M. $\square$

Theorem 4.5.2. *Let $\mathcal{X}$ be the set of minimal formations returned by the find-all-minimal-formations algorithm and $\mathcal{X}^{PSU}$ be the equivalent for the PSU extension. The following two conditions hold:*

(i) $\forall X \in \mathcal{X} \exists X^{PSU} \in \mathcal{X}^{PSU}$ such that $X \subseteq X^{PSU}$,
(ii) $\forall X^{PSU} \in \mathcal{X}^{PSU} \exists X \in \mathcal{X}$ such that $X \subseteq X^{PSU}$.

Proof. To prove (i), observe that for $X \in \mathcal{X}$, the MCF algorithm returns X when each player i chooses a seed strategy from X_i. This follows from the fact that minimal formations do not overlap (Benisch et al., 2006, Cor. 2). Therefore, directly from Theorem 4.5.1, there exists some X^{PSU} such that $X \subseteq X^{\text{PSU}}$.

To prove (ii), let $X^{\text{PSU}} \in \mathcal{X}^{\text{PSU}}$. Let X' be the minimum containing formation returned by MCF when each player i chooses a seed strategy from X_i^{PSU}. From Theorem 4.5.1, $X' \subseteq \mathcal{X}^{\text{PSU}}$. The set X' is a formation, therefore there exists some $X'' \subseteq X'$ that is a minimal formation. Hence, $X'' \in \mathcal{X}$ and $X'' \subseteq X^{\text{PSU}}$. $\square$

Under these rules, s_i can never be eliminated with the *optimistic partially-specified utility function*, when it would not have been with the fully-specified utility function. In addition, s_i cannot support the elimination of another strategy with the optimistic partially-specified utility function, when it would not have been supporting with the fully-specified utility function. Therefore, the each minimal formations is weakly contained within some formations found using the optimistic algorithm.

4.6 Finding Approximate Formations

Cheng and Wellman (2007) introduce a weakened version of the standard dominance condition, which they called δ-*dominance*. Applying this condition iteratively yields the set of strategies that survive iterative elimination of *weaker-than-weakly* dominated strategies (IE[wtw]DS). For equilibria found in the restricted game, the authors provide regret bounds on those profiles in the base game.

I define a similar condition for formations. Let $\widehat{U}_i$ be the function that specifies the best-response utility of player i for each $\sigma_{-i} \in S_{-i}$ when player i's strategy set is limited to X_i. Thus, $\widehat{U}_i(\sigma_{-i}; X_i) = \max\{u_i(s_i, \sigma_{-i}) | s_i \in X_i\}$. I extend the definition of regret to sets of strategies in the following way.

Definition 22 (Regret of a Strategy Set)**.** *Let $\emptyset \subset X_i \subseteq S_i$. The* regret *of player i for having the restricted strategy set X_i against $\Delta(X_{-i})$ is*

$$\delta_i(X_i | X_{-i}) = \max_{\sigma_{-i} \in \Delta(X_{-i})} \widehat{U}_i(\sigma_{-i}; S_i) - \widehat{U}_i(\sigma_{-i}; X_i).$$

I also define regret of a joint strategy set.

Definition 23 (Regret of a Joint Strategy Set)**.** *Let $\emptyset \subset X \subseteq S$. The* regret *over all players for having restricted joint strategy set X is $\epsilon(X) = \max_{i \in N} \delta_i(X_i | X_{-i})$.*

Using the set-wise extensions of regret, I define ϵ-formation.

Definition 24 (ϵ-Formation)**.** *A set of profiles $X \subseteq S$ is an ϵ-*formation *if $\epsilon(X) \leq \epsilon$. The set X is said to be an ϵ-*primitive formation *if no proper subset of X is an ϵ-formation.*

Theorem 4.6.1. *If X is an ϵ-formation at level $\gamma = \epsilon(X)$ and $\sigma \in \Delta^X$, then $\epsilon(\sigma|\Gamma) \leq \epsilon(\sigma|\Gamma_{S\downarrow X}) + \gamma$.*

Proof.

$$\begin{aligned}
\epsilon(\sigma|\Gamma) &= \max_{i \in N} \widehat{U}_i(\sigma_{-i}; S_i) - u_i(\sigma) \\
&= \max_{i \in N} [\widehat{U}_i(\sigma_{-i}; X_i) - u_i(\sigma) + \widehat{U}_i(\sigma_{-i}; S_i) - \widehat{U}_i(\sigma_{-i}; X_i)] \\
&\leq \epsilon(\sigma|\Gamma_{S\downarrow X}) + \gamma.
\end{aligned}$$

□

Aside from evaluating the regret of a set of joint strategies, we may be interested in finding a set of joint strategies with minimal regret (ϵ) under some budget k on the size of the profile space. Such a situation can occur when analysts are designing novel strategies from a set of existing strategies. New strategies are evaluated against a set of promising existing strategies using NE regret. To compute NE regret, we need estimates for the utility of all the profiles in the joint strategy space as well as the unilateral deviations to the new strategy. We may be limited in the number of observations we can make for these new profiles, due to limited computational resources. This implicitly bounds the size of the joint-strategy space we consider. I formulate the optimization problem as follows:

$$\begin{aligned}
\min \quad & \epsilon(X) \\
\text{s.t.} \quad & |X| \leq k.
\end{aligned} \tag{4.1}$$

Developing search strategies to find the optimal X is a fruitful area for future research. In order to solve the optimization problem, we need to efficiently compute $\epsilon(\cdot)$, which in turn depends on the calculation of $\delta(\cdot)$. We calculate $\delta_i(X_i|X_{-i})$ by determining the supporting strategies in $S_i \setminus X_i$. A strategy $\hat{s}_i \in S_i \setminus X_i$ *supports* δ if the maximum gain in utility $\tau > 0$ in the following linear program:

$$\begin{array}{rlcl} \max & \tau & & \\ \text{s.t.} & \sum \sigma_{x_{-i}} & = & 1 \\ & \sigma_{x_{-i}} & \geq & 0 \ (\forall x_{-i} \in X_{-i}) \\ & u_i(\hat{s}_i, \sigma_{-i}) - \tau & \geq & u_i(s_i, \sigma_{-i}). \end{array} \tag{4.2}$$

Notice that the last set of constraints in the linear program of (4.2) is slightly different than the set of constraints in the linear feasibility program of Algorithm 1. In Algorithm 1, we consider every $s_i \in S_i$ rather than just in X_i. The reason for the distinction is subtle. For a strategy to be included in a primitive formation it must be a best response to some mixture over X_{-i}, not simply an improving deviation, as in (4.2). The constraint in (4.1) causes us to loosen our criterion for accepting additional strategies in the ϵ-formation.

Let COMPUTE-TAU return the solution to the linear program of (4.2). The pseudocode for calculating $\delta_i(X_i|X_{-i})$ is given in Algorithm 3. For each $\hat{s}_i \in S_i \backslash X_i$, we determine the maximum τ and set δ to the maximum of those values. If $\tau \leq 0$, then $\hat{s}_i$ is *covered* by X_i. If $\tau > 0$, then $\hat{s}_i$ *supports* $\delta_i(X_i|X_{-i})$.

Algorithm 3 COMPUTE-DELTA($X_i, S_i, X_{-i}, u_i(\cdot)$)

```
$\delta \leftarrow 0$
for $\hat{s}_i \in S_i \setminus X_i$ do
    $\tau \leftarrow$ COMPUTE-TAU($\hat{s}_i, X_{-i}, u_i(\cdot)$))
    $\delta \leftarrow \max(\delta, \tau)$
return $\delta$
```

Given an efficient algorithm for determining $\epsilon(\cdot)$, we need an efficient search algorithm for identifying the optimal restricted game. Before discussing search algorithms, consider some general observations regarding $\epsilon(\cdot)$: ϵ is not monotonic in the subset relation, ϵ is not transitive, ϵ is not submodular or supermodular, and greedy selection of X is not optimal.

The two-player, three-action symmetric games of Table 4.1 highlights a few of these observations. Allowing for the abuse in notation, the regret in game (a) of choosing from the strategy set $\{A\}$ is 0, since all-A is a Nash equilibrium. Similarly, $\epsilon(\{A, B, C\})$ is 0, since all of the strategies are available. However, $\epsilon(\{A, B\}) = 1$ and $\epsilon(\{A, C\}) = 1$. Notice that $\epsilon(\{A\}) < \epsilon(\{A, B\}) > \epsilon(\{A, B, C\})$, which violates monotonicity and transitivity. Additionally, $\epsilon(\{A\}) + \epsilon(\{A, B, C\}) < \epsilon(\{A, B\}) + \epsilon(\{A, C\})$, which violates supermodularity. For game (b), $\epsilon(\{A\}) = 1$ and all other restricted games have a regret of 0. Therefore, $\epsilon(\{A\}) + \epsilon(\{A, B, C\}) > \epsilon(\{A, B\}) + \epsilon(\{A, C\})$, which violates submodularity.

(a)

	A	B	C
A	2	0	0
B	0	0	1
C	0	1	0

(b)

	A	B	C
A	-1	0	0
B	0	0	0
C	0	0	0

Table 4.1: Row player payoffs for games with non-modular $\epsilon(\cdot)$

The first three observations provide no effective bounds for branch-and-bound search. The fourth observation rules out a greedy search algorithm as optimal, however it may be a reasonable heuristic search method. Below I outline a simple algorithm to search over the join-semilattice of restricted games.

Algorithm 4 FIND-FORMATION(Γ, k, M)

```
$best \leftarrow null$
$queue \leftarrow$ Empty priority queue with maximum size $M$ ordered by $\epsilon(\cdot)$
ENQUEUE-INITIAL-GAMES($queue, \Gamma, k$)
while $queue$ is not empty do
    $game \leftarrow$ top($queue$)
    if $best$ is null or $\epsilon(best) > \epsilon(game)$ then
        $best \leftarrow game$
    ENQUEUE-CHILD-GAMES($queue, game, k$)
return $best$
```

Algorithm 4 uses two basic subroutines and a priority queue with a maximum size of M to determine the optimal restricted game for the given bound k. The two enqueue subroutines, ENQUEUE-INITIAL-GAMES and ENQUEUE-CHILD-GAMES, each present restricted games to the priority queue. The ENQUEUE-INITIAL-GAMES subroutine generates all restricted games where each player has a single strategy to choose from. The ENQUEUE-CHILD-GAMES subroutine generates all of the restricted games where some player has an additional strategy to choose from. For each generated restricted game in the enqueue subroutines, the regret of the game is calculated by the COMPUTE-EPSILON subroutine. The priority queue orders each restricted game by this calculated value.

Observe that the number of restricted games is exponential in the size of the players' strategy sets, therefore a finite (small) value for M is required for even small values of k and small games. Notice the special case of $M = 1$, which corresponds to simple greedy search. In the experiments of Section 4.7, I vary M and observe the effects on the regret of the selected restricted game.

4.7 Experiments

I consider two sets of experiments in evaluating the ϵ-formation finding algorithms. Note that both sets of experiments involve two-player games, and recall that formations and CURB sets are identical in this case. The first set considers a market game of interest, the TAC travel-shopping game (Wellman et al., 2007), and compares the restricted games given by *iterated weaker-than-weak dominance* (Cheng and Wellman, 2007) to those given by Algorithm 4. The second set, run on randomly generated games, investigates under what circumstances primitive-formation algorithms and ϵ-formation algorithms differ.

TAC Travel

Much like the TAC/SCM and TAC/AA games described in Chapter 1, the Trading Agent Competition Travel-Shopping game (TAC/Travel) is a simulation-based game representing a complex market scenario. The simulation is designed as an eight-player symmetric game. The payoffs are derived from Monte Carlo simulation (Wellman et al., 2007). Cheng and

Wellman (2007) consider a space of 27 candidate strategies using a two-player hierarchical reduction of the original eight-player scenario, forming a symmetric game with 378 distinct profiles. Using iterated elimination of strictly dominated strategies, the 27 candidate strategies are reduced to a set of 18 candidate strategies.

The authors investigate two greedy elimination algorithms by measuring the error of each algorithm after each iteration of the algorithm. The error is calculated as the largest regret with respect to the base game over all equilibria in the restricted game selected at the end of each iteration. Thus, ϵ is an upper bound on the error metric used by Cheng and Wellman. The two algorithms use *δ-dominance*, an approximate notion of dominance (see Definition 9), as a strategy elimination criterion. Both algorithms iteratively eliminate strategies until an error budget is reached. The error bound calculated by the algorithms is weakly monotonically increasing in the number of strategies eliminated; however, Cheng and Wellman provide a method for tighter bounds calculation that may reduce the error estimate but is no longer monotonic.

The results reported by the authors show that the greedy algorithms can form restricted games with 9-15 strategies that have an error bound of approximately 30, 6–8 strategies with an error bound of 10, 4–5 strategies with an error bound of 5, and 2–3 strategies with an error bound of 45. The actual error over these restricted games varies from 0 to approximately 20 in the case of the restricted games with 2–3 strategies.

When I apply the ϵ-formation algorithm to this dataset, some interesting features emerge. First, the smallest ϵ, single-strategy restricted game has a regret of 26.4, smaller than the bounds returned by the greedy algorithms of Cheng and Wellman for 2–3 strategy restricted games—the actual regret of the two-strategy restricted game is approximately 20. Second, there is a primitive formation that consists of two strategies. Therefore, we have a very aggressive pruning strategy that can prune all but two strategies and retain zero regret for the restricted game. In general, such an aggressive pruning will not result in a primitive formation, thus I analyze approximate formations in the experiments of Section 4.7.

Randomly generated games

The randomly generated games used this section are classified into two distinct types generated by GAMUT (Nudelman et al., 2005): *random* and *covariant*. The random class of games has payoffs that are uniformly and independently distributed in the range [-100,100]. The covariant class of games has payoffs that are distributed normally(0,1) with covariance r between players in a profile. For experiments in this section, I used a setting of $r = -\frac{1}{2}$ for covariant games. I generated 100 instances of each class with two players and ten strategies per player. I compute the regret for each pure-strategy profile in each instance. Figure 4.1 shows the empirical distribution of regret values over profiles in each game class.

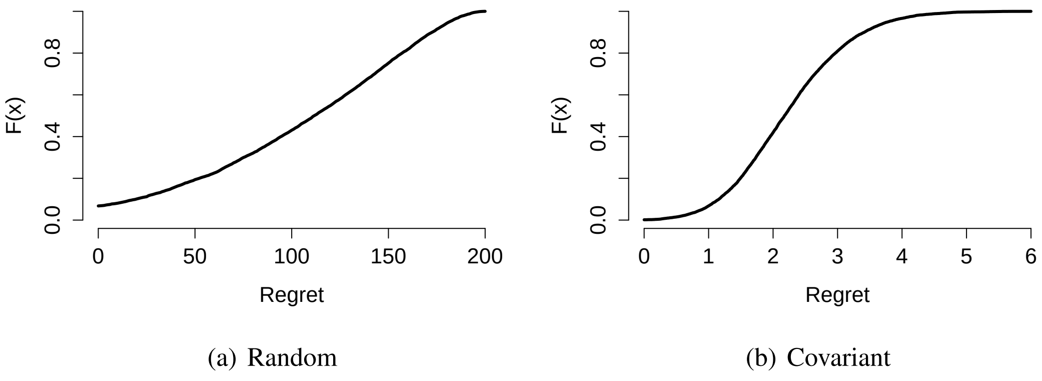
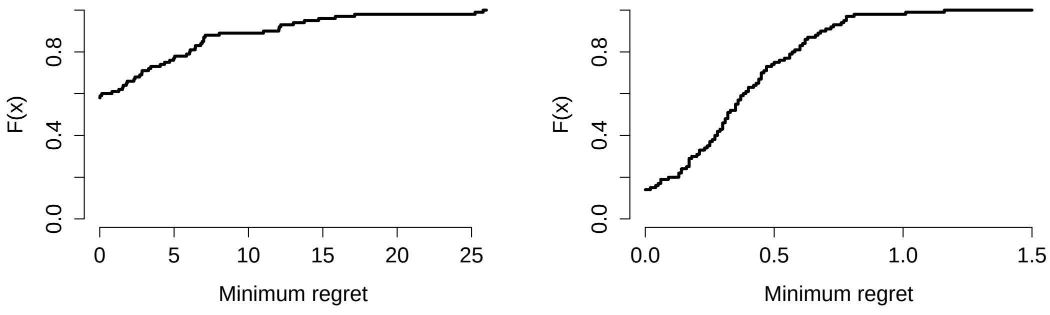


Figure 4.1: Empirical distribution function of regret values over profiles for random and covariant games.

The algorithms of Section 4.6 seek to find a restricted game whose regret is as small as possible for a given constraint k on the size of the game. Benisch et al. (2006) found experimentally that random games tend to have small smallest CURB sets (pure strategy equilibria), whereas covariant games tend to have large smallest CURB sets (nearly all strategies). While reviewing their findings, I make another observation. For those random games which do not have a pure-strategy NE, the smallest CURB sets tend to be large. This implies that we are unlikely to find minimal CURB sets for intermediate values of k unless there exists a pure-strategy Nash equilibrium.



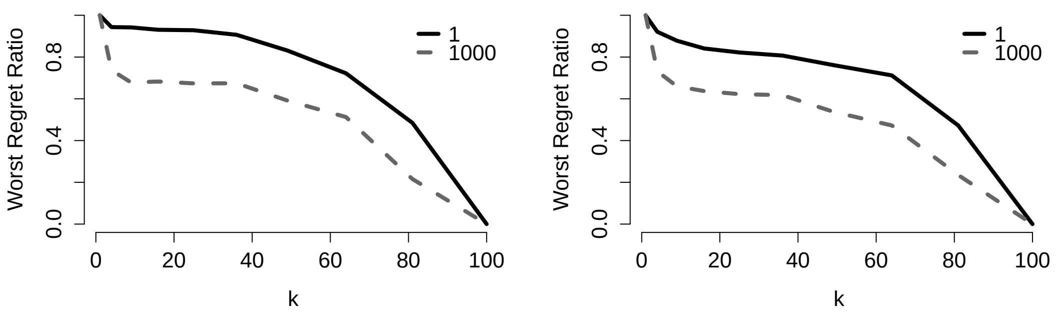
(a) Random (b) Covariant

Figure 4.2: Empirical distribution of minimum-regret values for random and covariant games.

Figure 4.2 shows the empirical distribution of minimum-regret values for pure-strategy profiles in the generated instances. GAMUT generated games with PSNE around 58% of the time for random games and 14% of the time for covariant games. Therefore with the same frequency, we can find minimal CURB sets where each player is selecting a single strategy. Thus, returning a restricted game consisting of the minimum-regret pure-strategy profile is optimal in nearly 58% and 14% of these games, respectively. Computing the minimum-regret profile is linear in the number of profiles and thus is accomplished efficiently.

The question remains as to what should be done in the remaining 42% and 86% of respective cases when we are bounded by an intermediate value for k. From Benisch et al.'s results, we can conclude that it is unlikely that a minimum CURB set, an ϵ-formation with $\epsilon = 0$, is found when k is much smaller than the size of the base game. However, using the algorithms of Section 4.6 we can find ϵ-formations where $\epsilon > 0$ for any k.

The number of restricted games is exponential and we have no known bounds relating the regret of different restricted games. However, the M parameter in Algorithm 4 allows us to explore some of the restricted games off of the greedy path if $M > 1$. Therefore, we remove instances from the two classes where a PSNE exists. On the remaining games, I run Algorithm 4 with two settings for M and various settings for k. For each game, we run the algorithm for each k where $k \in \{i^2 | 1 \leq i \leq 10\}$. Because the size of the base game is 100, the algorithm should return a game with $\epsilon = 0$ when $k = 100$, even in the greedy case. I use two settings for M; a setting of $M = 1$ corresponds to greedy search, whereas $M = 1000$ allows some non-greedy exploration. Maintaining a maximum queue size of 1000 allows a complete search of roughly the first two levels, or $k \leq 9$ in the generated instances.



(a) Random (b) Covariant

Figure 4.3: Worst-regret ratio for random and covariant games.

Figure 4.3 shows the results of running Algorithm 4 on the two classes of games, where the *worst-case regret ratio* is fraction of regret of the restricted game found by the algorithm when compared to the regret of the minimum-regret profile. Both the greedy ($M = 1$) and $M = 1000$ found restricted games with less regret than the minimum regret profile. In fact, when considering the random and covariant game classes, we can find a restricted game with worst-case regret of approximately 75% using only 9% of the base game space on average.

4.8 Discussion

Rational closure is a useful concept for selecting restricted games that capture relevant strategic information about the base game. In particular, primitive formations allow modelers to be confident that the regret of any profile calculated in the restricted game is the same as when calculated in the base game. I adapt the algorithms of Benisch et al. (2006) for calculating minimal CURB sets in two-player games to finding primitive formations in n-player games. In addition, I derive techniques for identifying primitive formations when utility functions are only partially specified, as is often the case in empirical game-theoretic analysis when the strategy space is large and sampling is costly. Finally, when primitive formations do not exist for a given restricted game size, I introduce ϵ-formations and an algorithm for finding them when faced with a constraint on the maximum size of the restricted game that is returned.

I show the ϵ-formation finding algorithm outperforms the iterated weaker-than-weak dominance strategy pruning algorithms on the TAC/Travel market game. In cases where the primitive formations of the base game are large, the ϵ-formation finding algorithm is able to decrease the regret of selected restricted games when evaluated over various classes of games. Future work should improve the ϵ-formation search algorithm by pruning irrelevant subtrees.

Chapter 5
Managing a Supply Chain with Trading Agents

Supply chain management (SCM) is concerned with the processes that govern the flow of materials and information in a network of business relationships (supply chain). As a field of study, SCM emerged from the study of inventory management systems in the mid-twentieth century and has produced numerous scholarly publications over multiple decades (Giunipero et al., 2008). The field has evolved from controlling the material flow from within an organization to managing materials and information across organizations. During this evolution, computer programs became critical to solving many of the problems posed in SCM from optimization to forecasting. This chapter is concerned with the design and analysis of the computer programs (trading agents) that implement the strategic behavior of the firms in a supply chain—such as procuring supplies and selling manufactured goods.

In the beginning, SCM's rapid adoption of computers drove the need for a new method of communication. In the late 1960s, a machine-readable form of electronic communication emerged. The term *electronic data interchange* (EDI) grew to encapsulate the transmission of this structured data (Bergeron and Raymond, 1992). EDIs allow business entities to transfer information within an organization or between organizations without a human intermediary. Thus, firms could now populate large databases of information automatically.

The vast stores of information provided new opportunities for resource management and strategic analysis, however exploiting these new found opportunities presented a difficult problem. Without a standardized (modular) system for managing this information, integrating new components into existing legacy systems can be excessively costly. Starting in the 1990s, *enterprise resource planning* systems (ERPs) arose to fill this need. Through standardization, ERPs allow organizations to integrate information systems—such as manufacturing, inventory, and accounting—that would otherwise be prohibitively complex.

Supply chain management leapt forward with the adoption of EDI and ERPs. These systems enable efficient execution of business plans by providing greater access to information. As SCM continues to evolve, we find that, increasingly, critical decisions are delegated to computer programs. *Mixed-initiative systems*—where both human and automated reasoning are employed—are being developed and studied in both industry and research communities. This chapter explores the design, development, and evaluation of systems that are *fully automated*. In this way, we view them as autonomous trading agents, functioning as the planner for the entire organization or node in a supply chain. At this level, the actions of the agents are inherently strategic.

Section 5.1 describes the scenario that I use for testing various strategies of agents. Because the TAC/SCM scenario is simulation based, we can employ the tools and techniques from previous chapters to analyze the strategic behavior of the agents. In Section 5.2, I review the general architecture of SCM agents that Ketter et al. (2008) found after surveying participants from various years in the competition. The analysis of Section 5.3 investigates progress of the field of agents in the TAC/SCM tournament. Section 5.4 describes the general structure of University of Michigan's agent, DeepMaize, and the design paradigm used to select DeepMaize for the tournaments.

5.1 TAC/SCM Scenario

This chapter investigates the strategic issues that arise from managing supply chains with trading agents. I pose the investigation as a case study of the Supply Chain Management game (TAC/SCM) introduced by Arunachalam and Sadeh (2005) and Eriksson et al. (2006). The TAC/SCM tournament—a supply chain management competition between agent designers—has been played annually since its inclusion in the Trading Agent Competition in 2003.

TAC/SCM is a dynamic supply chain environment where trading partners form short-term relationships. This is in contrast to the static, long-term relationships currently found in practice. This flexibility allows agents to explore trading and management strategies that are potentially more efficient. While obviously not of the highest fidelity, TAC/SCM captures many of the interesting strategic and non-strategic aspects of managing a supply chain, while making simplifications to keep a modest barrier to entry for potential research teams. Collins et al. (2004) provide a complete specification of the game. Further description and discussion is provided in many of the papers cited herein. Although most details of the game rules are inessential to the analysis here, I establish some context by providing a capsule description.

In TAC/SCM, six agents representing PC (personal computer) manufacturer agents compete to maximize their profits over a simulated year. There are 220 scenario days, and agents have approximately 14 seconds to make decisions each day. Agents participate simultaneously in markets for supplies (components) and finished PCs. There are 16 different types of PCs (divided into three market segments), defined by the compatible combinations of 10 different component types. Components fall into one of four categories: CPU, motherboard, memory, and hard disk. There are four types of CPUs and two types of each of the remaining components; one component from each category is required to produce a PC.

Agents negotiate deals with suppliers and customers through an *RFQ* (*request-for-*

quote) mechanism. The suppliers and customers execute policies defined by the game specification and implemented in the server. The suppliers have limited production capacity that varies during the game according to a random walk. They make offers and set prices based on their ratio of available capacity. The customer generates requests for PCs each day. The number of requests is driven by a stochastic demand process for each market segment.

Agents face substantial uncertainty in both markets. The underlying supplier capacities, customer demand parameters, and local state of other manufacturer agents are not directly observable, so agents must estimate these from other sources of information. There is also strategic uncertainty, since agents do not know the exact strategies employed by their competitors.

Each manufacturer is endowed with an identical factory that has limited production capacity, measured in *cycles*. Each PC type requires a different number of cycles to produce. Agents pay storage costs for all components and PCs held in inventory each day, and are charged (or paid) interest on bank balances. At the end of the game agents are evaluated based on total profit, and any remaining inventory is worthless.

5.2 Supply Chain Management Agents

TAC/SCM is a complex scenario and developing an agent is a correspondingly difficult task. On a given day in the TAC/SCM senario, an agent faces three types of decisions: *procurement*, *sales*, and *scheduling*. Procurement decisions consist of generating RFQs and accepting or rejecting offers from suppliers. Agents make sales decisions by determining the offer prices of RFQs received from customers. Finally, scheduling decisions include selecting which PCs to produce from available components, and planning delivery of the finished PCs to fulfill customer orders. These decisions take place under substantial uncertainty in the state of the markets. Predicting the state (both hidden and observable) is an important part of an agent's strategy.

Participating TAC/SCM teams have proposed various general architectures for agents. Ketter et al. (2008) distill the results of an informational survey regarding these architectures. Figure 5.1 gives a hierarchical interpretation of their findings. Top-level *features* include decision coordination mechanisms, choice of general optimization algorithms, and modeling choices for dealing with uncertainty. Under the umbrella category for decision coordination, agents specify procedures for dealing with inventory, procurement, sales, etc. For each of these subcategories, there is often a prediction component as well as a local optimization policy.

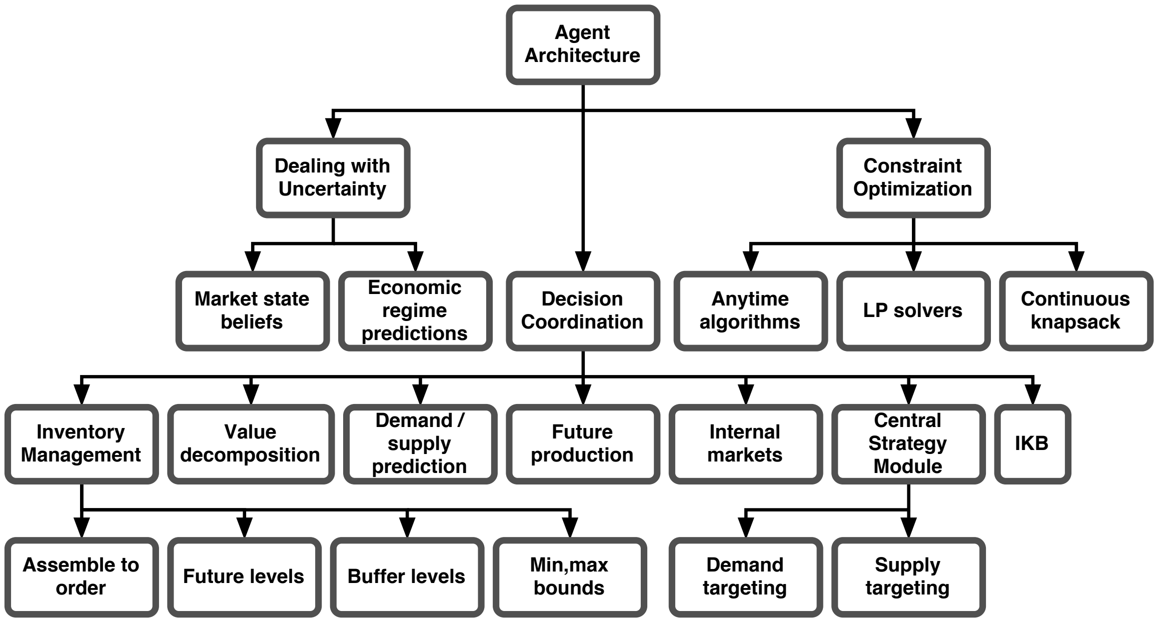


Figure 5.1: Features of agent architectures in TAC/SCM (Ketter et al., 2008).

There is anecdotal evidence to suggest that a majority of novel agent designs (architectures) in TAC occur in the first few years of the respective scenario. This is not to imply that subsequent years are inconsequential to research; rather, in the first few years we historically find incipient high-level designs—for instance, value-based decomposition (Kiekintveld et al., 2006) in TAC/SCM—and subsequent competitions' agents improve upon these designs with improvements to low-level components such as demand and supply prediction algorithms. Some low-level components of agent behavior that are common to most agents have inspired side competitions or challenges. In particular, the *TAC/SCM Prediction Challenge* (Pardoe and Stone, 2008) isolates the demand and supply price prediction problem agents face in TAC/SCM.

5.3 Measuring Progress

The main goal of this section is to determine whether agents are progressing—that is, whether subsequent years' agents are "better" than prior years. Agent progress supports the value of research competitions in tackling difficult problems like supply chain management; however, the multiagent nature of the scenario complicates progress metrics because agent *competence* is measured with respect the set of opponent agents' strategies that may change over time. I employ the strategy evaluation measures of Chapter 3 to show that, in fact, TAC/SCM agents have become better over time.

I center analysis around a TAC/SCM empirical game model. First, I introduce an empirical game model used to represent TAC/SCM and describe the process used to generate data for the model. The TAC/SCM scenario is implicitly defined by a server, which I use as the underlying simulator for the model. The strategy space is the set of agents submitted to the agent repository,[1] as well as candidates constructed for DeepMaize analysis. Table 5.1 lists the participating agents used in this analysis as well as their affiliations and descriptions, if available.

Following the description of the empirical game, I show the results of an isolated empirical game-theoretic analysis of the strategies for the 2005 and 2006 tournaments, respectively. In addition, I analyze the 2005 and 2006 strategies jointly. Evidence from the analysis supports the claim that agents are improving in competence over subsequent

[1]Designed and implemented by Joakim Eriksson (Swedish Institute of Computer Science) and Kevin O'Malley (University of Michigan), and available at `http://www.sics.se/tac/showagents.php`.

Agent	Affiliation	Description
Botticelli	Brown U	Benisch et al. (2004)
CMieux	Carnegie Mellon University	Benisch et al. (2009)
CrocodileAgent	University of Zagreb	Podobnik et al. (2006)
DeepMaize	U Michigan	Kiekintveld et al. (2006)
GoBlueOval	Ford Motor Co. and U Michigan	
Maxon	Xonar Inc.	
Mertacor	Aristotle U Thessaloniki	Kontogounis et al. (2006)
MinneTAC	U Minnesota	Collins et al. (2007)
PhantAgent	Politechnica U Bucharest	Stan et al. (2006)
SouthamptonSCM	U Southampton	He et al. (2006)
TacTex	U Texas	Pardoe and Stone (2007)

Table 5.1: TAC/SCM participants with affiliation.

tournament years.

Defining an empirical game model

The TAC/SCM server defines a simulation space with six players. The players' roles in the simulator are equivalent, thus forming a symmetric profile space. Given a symmetric game with N players and S strategies, there are $\binom{N+S-1}{N}$ distinct pure-strategy profiles. For example, in this analysis we may consider $S = 6$ agent strategies. With $N = 6$, this induces a total of 462 profiles that would need to be estimated for a full-granularity analysis. We can significantly decrease this number by restricting attention to cases where strategies are assigned to *pairs* of agents rather than individuals. Specifically, the resulting 3-player game, denoted SCM$\downarrow_3$, comprises only 56 profiles over the same 6-strategy set. The payoff to a strategy in an SCM$\downarrow_3$ profile is defined as the *average* payoff to the two agents playing this strategy in the original 6-player game.

In several contexts, it has been shown experimentally and theoretically that this form of *hierarchical game reduction* produces results approximating well the original unrestricted game, with great computational savings (Reeves, 2005; Wellman et al., 2005b). Although

I have not validated this specifically in TAC/SCM, intuitively I would expect this game to share the necessary property of payoffs smoothly varying with the number of other agents playing a given strategy.

Generating the observation set

I have collected results from well over 35,000 sample games combined in the TAC-05 and TAC-06 environments. We performed most of the simulations using a computing cluster operated by the University of Michigan. The cluster facility provides scalable and homogeneous processing, supporting parallel simulation with a fair allocation of computational power to each agent.

Each game simulation reserves seven CPUs for a period of one hour; one for each agent and one for the TAC/SCM server. We group games into sets of 3–5 to reduce the overhead cost of configuring the simulation on the assigned cluster nodes. Game results (in the form of server log files) are sent back to central repository. A central server tracks the results and submits new simulations to the cluster as each job is completed.

The simulated strategies potentially differ from the actual tournament agents in one important respect. Tournament agents can maintain state from one game instance to another, and so can *adapt* their strategy for later games based on experience in earlier games. Several agents take advantage of this opportunity, including the top-scoring agent from both 2005 and 2006 tournaments, TacTex (Pardoe and Stone, 2006). The simulation analysis is based on sampling from a pool of fixed strategies, so this necessarily restricts the agents to versions that adapt only *within* a game instance.

During a game simulation, much can go wrong, for example network outages or delays, or interference with one or more processors. We therefore attempt to filter our data set by removing game instances tainted in this way. We considered various procedures for identifying tainted games, ultimately settling on a very simple rule. A game is scratched if, for any agent, there are six or more days (out of 220) in which the server did not receive a message from that agent (as indicated by the game log).

Given the expense of generating samples by simulation (over 7 processor-hours per game), we seek to glean the most information we can from each data point. Toward that end, I employ the reduce variance techniques of Section 2.2. In the case of TAC/SCM, the most significant stochastic factor bearing on payoffs is the level of customer demand for PCs during the game.

SCM 2005

The TAC-05 analysis employs a dataset of 2,110 validated game instances, covering a minimum of 28 samples each of 56 distinct profiles of six TAC-05 agents: TacTex (Tx), Mertacor (Mr), DeepMaize (Dm), MinneTAC (Mt), PhantAgent (Ph), and GoBlueOval (Gb).[2] As shown in Table 5.2, the first four of these made it to the final round of the TAC/SCM-05 tournament (the other two finalists are not currently available in the repository), PhantAgent was a semifinalist, and GoBlueOval was a quarter-finalist.

Agent	Finals	Semifinals	Quarter-Finals	Seeding
TacTex (Tx)	4.74	3.57 [1]	17.78 [A]	14.89
SouthamptonSCM	1.60	4.62 [2]	3.50 [B]	10.05
Mertacor (Mr)	0.55	2.66 [2]	4.58 [B]	9.30
DeepMaize (Dm)	–0.22	3.68 [1]	17.49 [D]	10.23
MinneTAC (Mt)	–0.31	2.27 [1]	11.91 [A]	9.86
Maxon	–1.98	3.80 [2]	5.23 [C]	8.76
PhantAgent (Ph)	n/a	–6.64 [1]	7.03 [A]	9.87
GoBlueOval (Gb)	n/a	n/a	–2.60 [B]	12.60

Table 5.2: TAC/SCM-05 finalists, plus PhantAgent and GoBlueOval, with average scores ($M) from seeding through final rounds (semifinal and quarter-final groups in brackets).

Interaction among the strategies is one factor explaining differences in scores—and even relative rankings—between rounds of the tournament. Game-theoretic analysis serves

[2]The versions of these agents in the repository do not maintain state from game to game, so may differ from the actual tournament agents as noted above.

to assess the robustness of tournament rankings to strategic interactions. Another factor that explains differences between rounds is modifications to agents made between rounds (developers are allowed to modify agents between tournament rounds, but not within a round). In one case, both a semifinal and final round version of a single agent has been released, but we typically do not have access to all versions of the agent and are thus unable to investigate these variations in our empirical analysis.

The contour deviation graph of Figure 5.2 summarizes our stability analysis of the pure strategy profiles of the game. Each node represents a profile (three strategies). The outgoing edge from a node indicates the *best deviation* from that profile—that is, the transition providing the greatest gain in payoff for one agent switching strategies. For example, the profile with all DeepMaize (DmDmDm, in Level 4 around 10 o'clock) points to profile DmDmMr, which means that switching from DeepMaize to Mertacor in this context offers the greatest increase in payoff. That the arrow signifying the edge is solid rather than dashed means that the benefit is statistically significant in this case, at the $p \leq 0.05$ level.

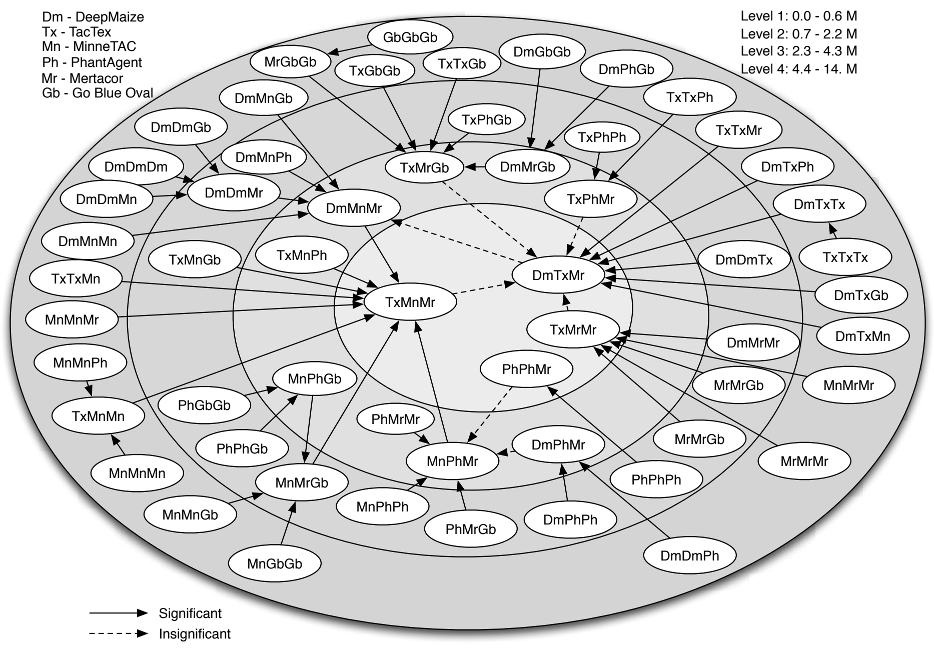


Figure 5.2: Deviation analysis for pure profiles of 2005 SCM$\downarrow_3$.

The magnitude of the potential benefit from deviating is represented by the node's placement in the diagram. The profiles in the innermost ellipse (Level 1) represent the most stable (closest to equilibrium), with $0.04M \leq \epsilon \leq 0.6M$. Concentric rings define levels with increasing values of ϵ. Level 4 (outermost ring) profiles are quite unstable, as a single agent (in the 3-player game) can benefit by at least 4.4M by deviating from its designated strategy. Note that the best deviation links usually, but not necessarily, connect profiles to more stable alternatives.

Since all profiles in Figure 5.2 have outgoing edges, we can conclude that the empirical game has no pure-strategy Nash equilibria (PSNE). Indeed, there exists a directed cycle among three relatively stable profiles, and all paths lead to this cycle.

There are, however, mixed strategy equilibria, and we have identified one symmetric Nash equilibrium, as well as several approximate equilibria. We found these mixtures using replicator dynamics (RD), and present them in Table 5.3. Specifically, we ran RD seven times; once with all strategies present, and once for each subset of five out of six. In all cases, the initial population is distributed uniformly. The profile generated by RD with all agents present is a symmetric Nash equilibrium. GoBlueOval is not played in this equilibrium, and indeed omitting that agent leaves the RD result unchanged. Two other RD results are approximate ($\epsilon < 1.0M$) equilibria; not surprisingly, these respectively omit the agents (DeepMaize and MinneTAC) with lowest positive probability in the known exact equilibrium.

The analysis reveals several striking observations. First, all agents perform quite poorly with many copies of themselves. Three out of the four most unstable profiles (MnMnMn, TxTxGb, TxTxTx, and MrMrMr, respectively) comprise a single strategy. This fact can be explained by the multiple copies all competing for the same "niche", or exploiting opportunities typically left available by other agents (but not themselves, of course). In addition, some of the problem may be simply that the agents are hardwired to procure components on certain days or with certain lead times, and these naturally interfere when more than one

Agent	all	(DeepMaize)	(TacTex)	(MinneTAC)	(PhantAgent)	(Mertacor)	(GoBlueOval)
DeepMaize	.055	—	.015	.035	.219	.326	.055
TacTex	.112	.137	—	.100	.210	.156	.112
MinneTAC	.057	.079	.106	—	0	.109	.057
PhantAgent	.400	.418	.533	.482	—	.271	.400
Mertacor	.376	.366	.346	.384	.559	—	.376
GoBlueOval	0	0	0	0	.012	.138	—
ϵ	0	0.49M	1.46M	0.42M	1.28M	3.50M	0

Table 5.3: Profiles resulting from replicator dynamics. Each column presents probabilities for a mixed profile, with associated ϵ in SCM$\downarrow_3$ specified in the bottom row. The first column presents the result from RD including all agent strategies (initial proportions uniform). Subsequent columns respectively omit one strategy from the RD process.

copy exists. Similarly, multiple copies may make the same predictions and estimates of prices and other market conditions; thus, they may be making bidding and other decisions in an interfering manner.

Second, PhantAgent performs much better in the game-theoretic sense than might be expected from the TAC/SCM-05 tournament outcome.[3] PhantAgent is least sensitive to playing with copies of itself, and appears with substantial probability in all the profiles produced by RD in Table 5.3. In fact, it is most probable in all but two cases: the one where it was excluded, and the one with highest ϵ value.

Third, Mertacor appears especially strong in a wide variety of contexts. Like PhantAgent, Mertacor is present with large probability in all the symmetric stable profiles identified. Most remarkable is that of the 35 profiles without Mertacor, 30 of them have a best deviation where some strategy changes to Mertacor. Of the 21 profiles with Mertacor, the best deviation changes from Mertacor in only three. In addition, Mertacor is also a best

[3]We are unaware of specific problems that may have afflicted the agent in the semifinal round, but this is a possibility.

response to players playing uniformly at random.

The equilibrium analysis complements the regret analysis shown in Table 5.4. Due to the large set of support, the NE regret measure does little to distinguish the strategies, however we do get additional insight from the max-regret values. The max regret—the maximum over all possible opponent profiles—of each strategy is fairly large. All agents have situations where they perform poorly, supporting the conclusion that, of these strategies, there are no approximately dominate strategies.

Agent	NE Regret	Max Regret
DeepMaize	0	6.33
TacTex	0	9.66
MinneTAC	0	13.89
PhantAgent	0	6.61
Mertacor	0	7.69
GoBlueOval	0.22	6.13

Table 5.4: TAC/SCM 2005 tournament strategy comparison using stability metrics ($M).

I note that the first observation above raises some questions about this analysis approach. Presumably TAC entrants design their agents with tournament play in mind, and so may not be concerned about the performance of their agents with copies of themselves in the environment. The restricted-game analysis is especially sensitive to this question, since all profiles have at least two copies of any strategy present. On the other hand, one might argue that performance in self-play is important, and the tournament unduly neglects this aspect of strategy. Furthermore, because the game logs provide a detailed account of observable agent behavior, mimicking this behavior is relatively simple along some dimensions of an agent's strategy, such as early-game procurement. Therefore it is reasonable to assume that an agent could face behaviorally similar competition agents and should design with self-play performance in mind.

SCM 2006

Agent	Finals	Semifinals	Quarter-Finals	Seeding
TacTex (Tx)	5.85	7.55 [2]	7.48 [B]	13.73
PhantAgent (Ph)	4.14	5.71 [2]	17.37 [C]	12.56
DeepMaize (Ds,Df)	3.58	6.46 [1]	9.61 [A]	16.60
Maxon	1.75	4.08 [1]	17.74[D]	10.63
Botticelli	0.48	1.94 [1]	0.83 [A]	4.21
MinneTAC (Mt)	–2.70	2.06 [2]	13.45 [C]	9.59

Table 5.5: TAC/SCM-06 finalists, with average scores ($M) from seeding through final rounds (semifinal and quarter-final groups in brackets).

The agents that competed in the 2006 TAC/SCM finals are listed in Table 5.5. Versions of five of these agents were released to the agent repository: TacTex (Tx), PhantAgent (Ph), DeepMaize (Ds and Df), Maxon, and MinneTAC (Mt). Two versions of DeepMaize were released, corresponding to versions that played in the final round and semifinal round (significant changes were made to the agent between rounds, particularly to procurement behavior). The MinneTAC agent is the version that played in the semifinal round. This agent was also changed for the final round, but the final round version has not been released. Both semifinal and final round versions of Maxon were released. We analyze five of the seven agents available, including both versions of DeepMaize but excluding both Maxon agents.[4] The full symmetric game for the five agents we include in our analysis comprises 35 profiles. We have over 1,100 validated game instances, with a minimum of 15 samples for each profile (typically 30 or more).

Profile stability results for the 2006 agent set are shown in Figure 5.3. This game contains a pure strategy Nash equilibrium (DsPhTx) and an approximate equilibrium (DsDsTx) that has a small, statistically insignificant, benefit of $0.09M$ for deviating to the PSNE. We also applied replicator dynamics to this game to search for symmetric mixed equilibria,

[4]Maxon was the last agent to be released, and we do not have enough simulation data for these agents to be included in our full analysis.

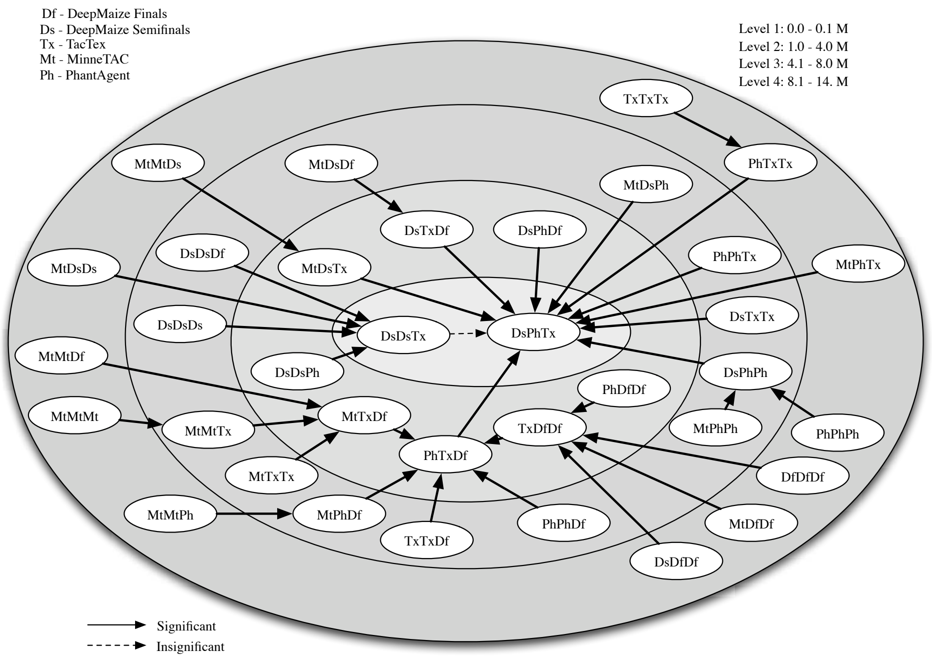


Figure 5.3: Deviation analysis for pure profiles of 2006 SCM$\downarrow_3$.

starting from mixtures generated uniformly at random. In all cases, RD converged to a mixture of TacTex, PhantAgent, and DeepMaize SF. Figure 5.4 shows the field for replicator dynamics over the simplex of these three strategies. The fixed point corresponds to the symmetric Nash equilibrium mixture (0.254, 0.188, 0.558).

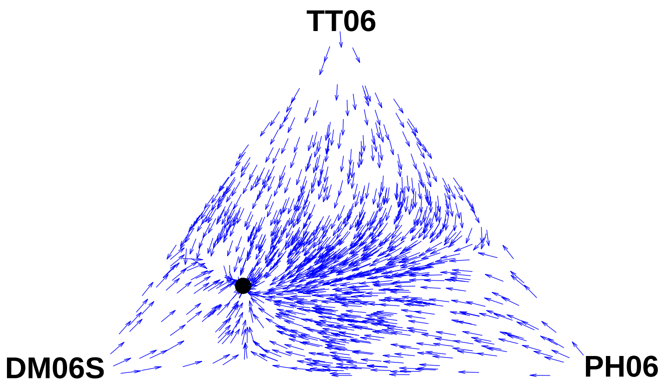


Figure 5.4: Replicator dynamics field for the top three agent strategies from the 2006 SCM$\downarrow_3$. The Nash equilibrium (0.254, 0.188, 0.558) is shown as a black dot.

Table 5.6 presents several statistics about the deviations in 2006 SCM$\downarrow_3$. *Percent positive deviations* is the fraction of possible deviations to the agent that result in a net benefit. *Best Deviation* is the number of instances where deviating to the agent was the most beneficial deviation. *Mean Deviation* and *std. error* reflect the average benefit ($M) for deviating to this agent, which may be negative. Deviations to TacTex, PhantAgent, and DeepMaize SF are beneficial in at least 60% of the cases. The mean value for deviating is highest for TacTex and DeepMaize SF, and TacTex is the best deviation most frequently. The three agents comprising the PSNE are nearly indistinguishable in this analysis.

Agent	% Positive Deviations	Best Deviation	Mean Deviation	Std. Error
TacTex	61.67	18	1.45	5.91
DeepMaize SF	63.33	5	1.43	4.68
PhantAgent	60.00	8	0.89	4.77
DeepMaize F	53.33	3	0.88	4.62
MinneTAC	11.67	0	–4.67	6.41

Table 5.6: Deviation statistics for agent strategies of 2006 SCM$\downarrow_3$.

Table 5.7 presents regret statistics for the agents in 2006 SCM$\downarrow_3$. NE regret is calculated with respect to the equilibrium given in Figure 5.4.

Agent	NE Regret	Max Regret
TacTex	0	10.36
DeepMaize SF	0	12.11
PhantAgent	0	11.42
DeepMaize F	0.89	10.04
MinneTAC	4.06	23.79

Table 5.7: TAC/SCM 2006 tournament strategy comparison using stability metrics ($M).

Perhaps the most striking result is that the semifinals version of DeepMaize clearly outperforms the finals version in this game-theoretic environment. This is another instance where an agent that did not participate in the finals shows strong performance in our game-theoretic analysis. Given these results, the selection of the weaker version of DeepMaize to play in the final round would have been prevented. In the combined analysis, I discovered further evidence that given only the 2005 agent strategies and the two versions of DeepMaize, the semifinals version is a more robust choice. This suggests that the type of analysis I present here should have applications to strategy selection as well as post-tournament analysis. Finally, notice the general increase in max regret across agents. I observe that agents in the 2006 game show similar difficulties playing against copies of themselves to the 2005 agents; all of the profiles with six identical agents are among the least stable profiles again. Self-play seems to be a major contributer to the increase in max regret. This effect may also partially explain why the profile with the top three agents playing is a PSNE. None of the agents has enough of an advantage over the others to overcome the penalty from playing against more copies of itself, so none of the deviations to these three is beneficial.

Combined 2005 & 2006 Analysis

The previous analysis focused on analyzing sets of agents from a single year of competition. One of the exciting opportunities afforded this empirical analysis methodology is to consider combinations of agents not observed during tournament play, including agents from different years. To facilitate general comparisons between agents, we use the NE-response ranking of Section 3.3.

For this analysis, I employ data from approximately 10,000 simulations focused on the profiles containing eleven agents from 2005 and 2006 (listed in Table 5.9). The experiments using the top 2005 agent sets as background context were simulated using both the 2005 and 2006 server rules. The simulations using the top 2006 agents as background context were run using only the 2006 server rules, which have two modifications from the 2005 rules. Under 2006 rules, the identity of opposing agents is revealed at the start of the game, and the reputation mechanism for suppliers was slightly modified. Agents from 2005 are compatible with the 2006 server, but may be at a slight disadvantage because they were not designed for the new rule set.

I begin by testing the hypothesis that the strongest agents from 2006 should show substantial improvements over the strongest agents from 2005. The first step is to select a symmetric Nash equilibrium for the 2005 agent set {DeepMaize, Mertacor, PhantAgent, TacTex}, which corresponds to the support of the mixed strategy equilibrium of the full 2005 game minus MinneTAC.[5] Using this 2005 equilibrium as the background context, we test possible deviations to three of the top 2006 agents. The results are given in Table 5.8, along with the background context. Each of the 2006 agents is a beneficial deviation from the 2005 equilibrium using both the 2005 and 2006 server rules, offering strong support for the hypothesis of improvements from 2005 to 2006. In the subgame that includes the background agents and DeepMaize SF, DeepMaize SF is the sole survivor of iterated elimination of dominated strategies, providing even stronger evidence for improvement in this agent.

Background Context		Deviation Gain (ϵ)		
05 Agent	Mixture		Server Rules	
DeepMaize	0.083	06 Agent	2005	2006
Mertacor	0.431	PhantAgent	5.33M	6.57M
PhantAgent	0.314	TacTex	5.07M	4.73M
TacTex	0.172	DeepMaize SF	4.22M	4.56M

Table 5.8: Deviation gain comparison of top 2006 agents in the context of a symmetric mixed Nash equilibrium of top 2005 agents for the 2005 and 2006 server rules.

In Table 5.9 I present a ranking of eleven agents from 2005 and 2006 in the context of the symmetric mixed Nash equilibrium given in Figure 5.4.[6] This equilibrium is robust to the addition of the 2005 agents into the strategy pool. All agents are ranked based on their NE regret with respect to the equilibrium context. This ranking is interesting par-

[5]This omission does not change the context substantially, and requires much less data.

[6]This is the only symmetric equilibrium we have found after extensive search, but we cannot guarantee that it is unique.

ticularly because it spans agents that have never faced one another directly in tournament competition. Table 5.9 also gives the tournament results, where applicable.

Agent	NE Regret	Tournament Scores Finals 05	Finals 06
TacTex 06	0	n/a	5.85
PhantAgent 06	0	n/a	4.15
DeepMaize 06 SF	0	n/a	n/a
Mertacor 05	0.57	0.55	n/a
DeepMaize 06 F	0.95	n/a	3.58
Maxon 06 S[7]	1.03	n/a	n/a
MinneTAC 05[7]	1.23	–0.31	n/a
PhantAgent 05	1.51	n/a	n/a
DeepMaize 05	3.18	–0.22	n/a
MinneTAC 06	3.48	n/a	–2.70
TacTex 05	5.96	4.74	n/a

Table 5.9: Ranking of eleven TAC/SCM agents based on deviations from an equilibrium context, along with tournament results (in \$M).

This ranking supports the case for substantially improved agent performance in the 2006 competition. Deviating to a 2005 agent from the 2006 equilibrium typically incurs a large loss. The exception is Mertacor 05, which shows a relatively small loss—smaller than two of the 2006 agents, including DeepMaize F which placed third overall. This agent continues to show strong performance in the game-theoretic analysis. I also note that this ranking generally corresponds to the ranking based on tournament results. The exception is TacTex-05, which ranks lower than one might expect based on tournament performance.

5.4 Designing DeepMaize

The goal in this section is to establish a principled development methodology for DeepMaize, the University of Michigan's TAC/SCM agent. The methodology focuses on an

[7]These strategies had a lower minimum number of samples, 13 & 18, respectively, than the remaining nine strategies.

iterative approach of introducing novel strategy features; then, evaluating and tuning them through empirical game-theoretic analysis. This approach is empirically validated by a first-place finish in the 2008 and 2009 TAC/SCM tournaments as well as improved NE-regret and max-regret values.

Developing an agent strategy for a TAC scenario can be a complex task—for example, the candidate strategy chosen for DeepMaize 2008 has 11,647 lines of code in 192 classes, not including a multitude of associated libraries. The development team has included over 10 contributing developers and researchers over its seven-year existence. The agent architecture for DeepMaize is defined by a set of modules and corresponding parameters (Kiekintveld et al., 2006). A concerted effort is made by the developers to expose *strategically relevant* behavioral parameters for the agent that can be tuned by empirical game-theoretic analysis.

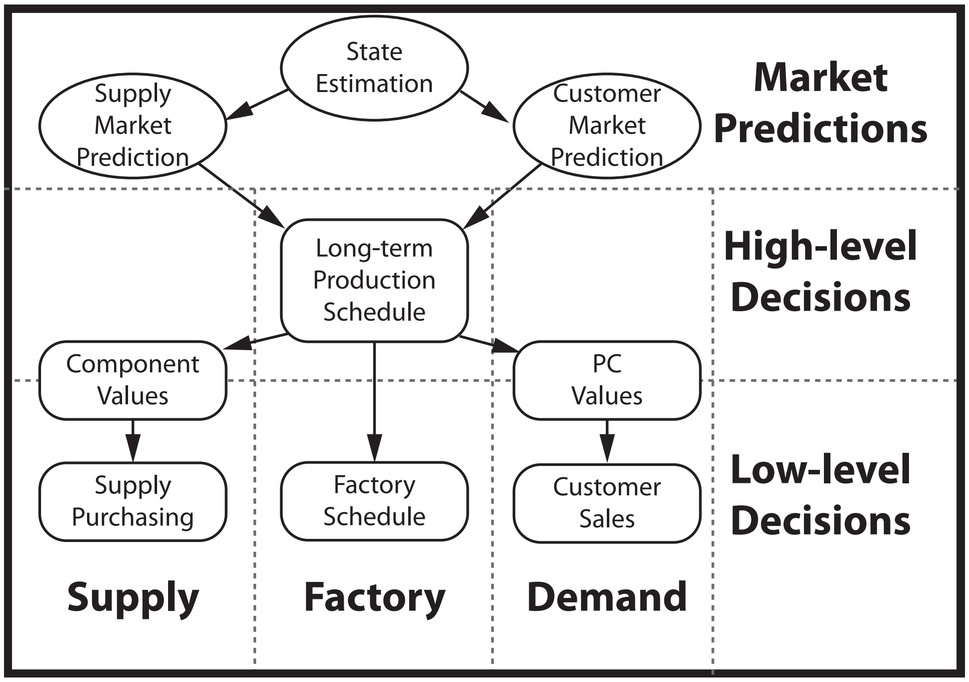


Figure 5.5: DeepMaize decision process

Figure 5.5 depicts the relationship between the modules. The architecture of DeepMaize has two main categories: *prediction modules* and *decision modules*. DeepMaize predicts various pieces of market information based on observations of supply (component) and demand (PC) prices and quantities. Manufacturers see component offers (prices) from suppliers for their own RFQs. Figure 5.6 shows an example of a supplier price curve. Each day DeepMaize must infer changes in whole curve, while only receiving price information for at most five points on the curve. For the PC market, DeepMaize predicts the probability that an offer will be accepted given a specific bid (sometimes called the probability of *win-given-bid*), based on past acceptance notification from customers. Kiekintveld et al. (2009) describe in detail the market prediction algorithms implemented by DeepMaize.

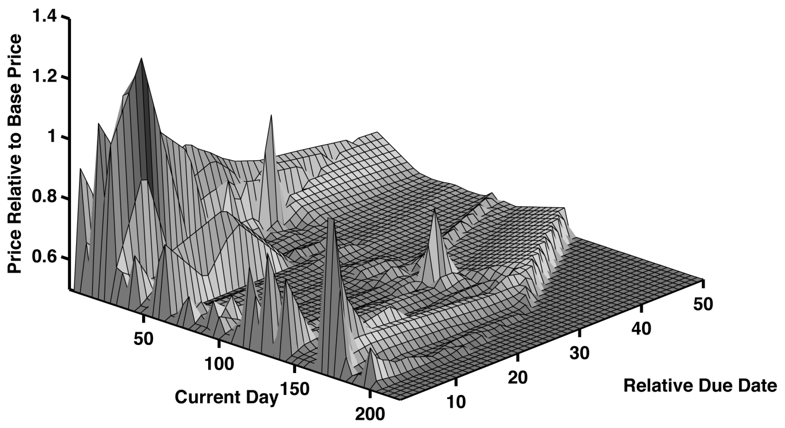


Figure 5.6: Example supplier price curve

Using various market state and price forecasts, DeepMaize creates estimates of the marginal value of resources through its *long-term production scheduler*. These values coordinate decisions across all low-level decision modules: *supply*, *factory*, and *demand*. These low-level modules make use of the market state predictions, however this value-based decomposition allows each module to optimize independently.

In what follows, I describe the regret-based development approach used by the DeepMaize team to select the agent (strategy) used in the given year's tournament from a set of candidate strategies. The process is iterative and centers around the construction of novel strategies that refute the current equilibrium; in other words, strategies that are positive deviations from an equilibrium in the restricted game. Unlike Section 5.3, which considers strategies at an atomic level, in this section we specify candidate strategies by component and parameter settings. In many cases, we construct novel candidate strategies by varying components or parameter settings from existing agents or promising candidates. For instance, two candidate strategies may be identical except for the amounts of initial components requested at the beginning of a simulation. The choice of a compact parameter set is important for empirical game-theoretic analysis to be feasible.

The basic design process proceeds as follows:

1. Select stable background context (equilibrium of restricted game),
2. Create novel strategies from stable strategies by varying component or parameter settings,
3. Simulate deviations from background context.

During the design of DeepMaize we often test multiple *features*—parameterizations or component decompositions—in parallel. The strategies testing a given feature are isolated in a restricted game. Promising candidates are then merged into a larger restricted game for analysis.

2007 Candidates

DeepMaize 2007 was selected after performing a series of experiments. Each experiment evaluated a set of candidate strategies on a particular component or parameter setting. The candidate strategy descriptions are given in Table 5.10, following which I review the results of the experiments.

The first experiment determined the effect of the various day-0 procurement parameterizations. Over the history of TAC/SCM tournaments, determining how much to order on the first day of the game has been a critical strategic problem faced by agent designers. Wellman et al. (2005a) show that it was perhaps *the* most critical aspect of agent behavior in the early tournaments. Although the game designers have reduced the strategic importance of initial procurement through various mechanisms, day-0 procurement remains a key component of an agent's strategy. We created four *candidate* variants of the DeepMaize 2006

Candidate ID	Description
C07-8	Major code update from DeepMaize 2006 SF with day-0 parameterization
C07-9	C07-8 update with shorter day-0 durations
C07-10	C07-8 with slightly longer day-0 durations
C07-11	C07-8 with longer day-0 durations
C07-12	C07-8 with new scheduling/valuation policy
C07-33	C07-12 with new prediction models, more aggressive in low demand
C07-34	C07-33 with restricted end-game risk-taking, lower/faster long-term

Table 5.10: TAC/SCM-07 candidate DeepMaize strategies.

semifinals agent, labeled C07-8 through C07-11. The parameter descriptions are given in Table 5.10; conceptually, the four vary according to day-0 aggressiveness with C07-9 being the most aggressive (ordering more, earlier). We simulated a restricted game consisting of the four candidate strategies. The results are given in Table 5.11.

Agent	NE Regret	Max Regret
C07-8	0.60	2.27
C07-9	0	1.25
C07-10	0	3.12
C07-11	0.59	2.66

Table 5.11: Stability metrics ($M) for candidates C07-8 through C07-11 in the *day-0 parameter test*.

The aggressive candidate, C07-9, has the minimal max regret and is part of the sample equilibrium returned by a sample run of replicator dynamics. Varying the aggressiveness in the range tested seems to have only a mild impact on scores (0.6M). Using C07-09 as a basis, we created candidate strategies with a new scheduling/valuation policy (C07-12); new prediction models and more aggressive behavior in low demand situations (C07-33); and with reduced end-game risk-taking behavior and with a lower quantity, faster purchasing long-term procurement policy (C07-34). These candidates were selected after a series of experiments, each varying an isolated aspect of behavior.

Agent	NE Regret	Max Regret
C07-9	0.20	3.40
C07-12	0.70	3.18
C07-33	0.80	3.06
C07-34	0	2.24
PH06	0	11.00
TT06	0	14.78

Table 5.12: Final 2007 candidate comparison using stability metrics ($M).

We simulated a large 7-strategy restricted-game with the strategies listed in Table 5.12. Starting from a uniform distribution, the Nash equilibrium returned by replicator dynamics had large support (98%) in C07-34 and small support (1%) in strategies PH06 and TT06, respectively. The C07-34 candidate also had the lowest max regret with $2.24M. We chose candidate C07-34 as the DeepMaize 2007 finals strategy. The full 2007 development tree is given in Figure 5.7.

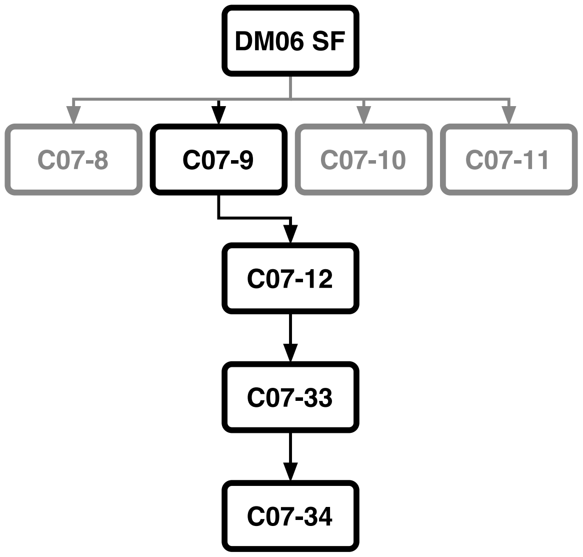


Figure 5.7: Development tree for DeepMaize 07 candidate strategies.

Agent	Finals	Semifinals	Quarter-Finals	Seeding
PhantAgent	8.67	10.38 [2]	15.70 [B]	13.30
TacTex	6.31	5.75 [2]	13.61 [C]	12.65
DeepMaize	5.45	9.759 [1]	21.10 [A]	17.76
Maxon	1.79	5.631 [1]	10.46 [C]	13.15
Tinhorn	1.34	6.94 [1]	19.36 [A]	10.24
CMieux	1.24	2.66 [2]	11.70 [C]	7.072

Table 5.13: TAC/SCM-07 finalists, with average scores ($M) from seeding through final rounds (semifinal and quarter-final groups in brackets).

The agents that competed in the 2007 TAC/SCM finals are listed in Table 5.13. During the semifinals, we noticed increasingly aggressive *early-game long-term procurement* by PhantAgent. Early-game long-term procurement is a related to day-0 procurement, but has distinguishing characteristics. In the initial phase of orders (day-0 procurement), agents submit RFQs for *early* to *mid*-range arrival dates. These orders are often expensive as suppliers rush to fulfill them with limited capacity; however, the profits made on the PCs assembled from these early arrivals can be large because an agent may have a short-term (near) monopoly on corresponding PCs. Conversely, in the early portion of a game the prices of components ordered further into the future (long-term) are generally low relative to the prices at which they can be fulfilled later in the game. Agents can exploit this by submitting orders early on for components that should arrive much later in the game. This is risky, as agents may order more components than they actually need if demand for PCs is lower than expected. At the time, we did not have enough computational resources to fully investigate the implications, which turned out to have dire consequences. Both PhantAgent and TacTex increased their aggressiveness in the finals. Largely due to this change in behavior, they finished first and second, respectively.

After the tournament, versions of three of these agents were released to the agent repository: TacTex (Ts and Tf), PhantAgent (Ph), and DeepMaize (DM). Two versions of TacTex were released, corresponding to versions that played in the final round and semifinal round.

Table 5.14 gives the results of a 9-strategy restricted-game analysis of the DeepMaize, PhantAgent, and TacTex 2007 tournament agents, as well as, top-performing agents from prior years. A sample equilibrium determined through replicator dynamics has support that consists of DM07 F, PH07, and TT07 F—the three highest-scoring agents of the 2007 tournament finals. The three DeepMaize strategies have the lowest max-regret in the analysis. In particular, the DeepMaize 20007 finals strategy has a very low max-regret value, implying that players can at most expect to lose \$2.63M compared to the best strategy for any tested situation.

Agent	NE Regret	Max Regret
DM07 S [C07-9]	0.32	3.40
DM07 F [C07-34]	0	2.63
PH07	0	48.84
TT07 S	2.90	16.95
TT07 F	0	10.89
DM06 S	3.21	8.17
PH06	1.31	11.00
TT06	1.03	14.78
MR05	2.98	14.67

Table 5.14: Final 2007 tournament strategy comparison using regret metrics (\$M), with accompanying agents from prior years.

2008 Candidates

The early-game long-term procurement strategy introduced by PhantAgent in the 2007 tournament exposed an interesting and valuable component of strategic behavior in TAC/SCM. Moreover, this aspect of their behavior was observable and easily replicated. Many of the teams participating in the 2007 tournament noticed the aggressive behavior and began experimenting with variations of PhantAgent's early-game long-term producement strategy in their agents for 2008.

While not completely isolating the effect of early-game long-term procurement strategy, Figure 5.8 shows the finals scores for DeepMaize, TacTex, and the range of the other finalists over consecutive tournament years since 2006. In 2007, PhantAgent beat both TacTex and DeepMaize. Additionally in 2007, there was a general increase in scores received by all of the finals agents (highlighted by the gray box), largely due to the aggressive early-game behavior of PhantAgent and TacTex. While having a few aggressive agents in the tournament seemed to increase scores (especially for the aggressive agents), there is a point at which adding aggressive agents is detrimental to all agents—as shown by the decline of "other" agents' scores in 2008 and 2009. This evidence suggests that early-game long-term procurement is a key strategic issue in agent design for TAC/SCM.

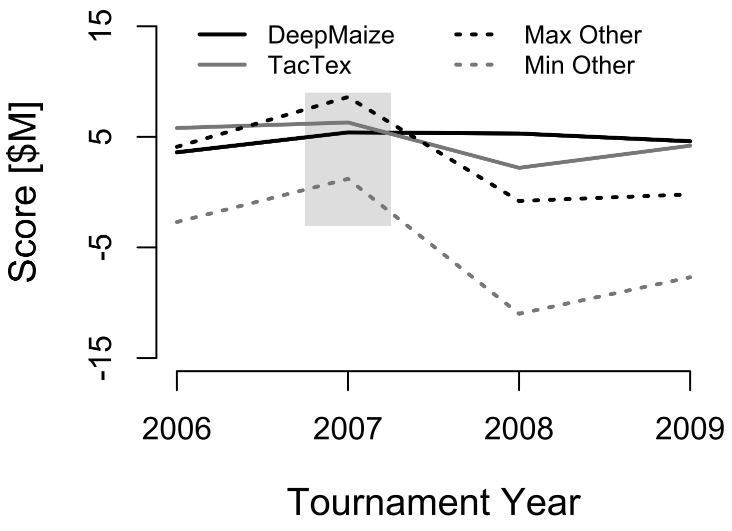


Figure 5.8: TAC/SCM tournament finals scores

As with 2007, DeepMaize 2008 was selected after performing a series of experiments. The 2008 candidate experiments evaluated various predictors, early-game long-term procurement strategies, and (PC) bid improvement algorithms. The full feature configurations of the candidate strategies (C08-$\{0,\cdots,29\}$) are given in Table 5.15.

	Predictions					*Controller*					
	Customer Dataset		Component Horizon Treatment			Bid Improvement			EG Procurement		
ID	SCM05	SCM[06-07]	AIO	INTRP-Bug	INTERP	EQ	SA	GA	07	PH	07+
0	✓		✓			✓			✓		
1	✓			✓		✓			✓		
2		✓	✓			✓			✓		
3-5		✓		✓		✓			✓		
6		✓		✓		✓				✓	
7		✓		✓		✓					✓
8		✓		✓			✓			✓	
9		✓		✓				✓		✓	
10	✓	✓		✓		✓				✓	
11	✓				✓	✓			✓		
13		✓			✓			✓			✓
14		✓			✓	✓				✓	
15		✓	✓			✓				✓	
17-27		✓		✓		✓				✓	
28-29		✓			✓	✓				✓	

Table 5.15: DeepMaize 08 tested feature matrix.

The first experiment determined the effect of the new component and customer predictors. DeepMaize uses both online and offline learning for its predictors. The datasets used for offline learning are extracted from game logs created by tournament and experimental play. We used the 2007 DeepMaize finals agent and created new datasets from the 2006 and 2007 tournaments, as well as introduced a new type of component price predictor. C08-1 differed from C08-0 on the component price predictor and C08-2 differed from C08-0 on the customer prediction dataset. The new component price predictor used by C08-1 contained a bug. We simulated a restricted game consisting of the three candidate strategies. The game has multiple pure-strategy Nash equilibria, however the mixed-strategy Nash equilibrium returned from replicator dynamics (initially uniform over strategies) has support that consisting of C08-0 and C08-2 only. The max-regret calculation (see Table 5.16) shows C08-0 to have a much lower regret (approximately half) that of C08-1 and C08-2. Based upon this results, we created candidate C08-3 as a variation of the C08-0 and C08-2 candidates, with a different dataset used to train the component price predictors. Starting with candidate C08-5, we used a variation of C08-3 that uses multiple component price predictors for different respective prediction horizons; it is similar in flavor to the predictor used by C08-1, except with better prediction accuracy.

Candidate	NE Regret	Max Regret
C08-0	0	0.86
C08-1	0.31	1.59
C08-2	0	1.83

Table 5.16: Initial 2008 component and customer prediction candidate comparison using stability metrics ($M).

The next experiment varied the early-game long-term component procurement policy of the candidates. Candidate C08-5 uses the DM07 F procurement policy, candidate C08-6 uses the PH07 procurement policy, and candidate C08-7 uses an enhanced version of the DM07 F policy. We included candidate C08-0 as a baseline in a 4-strategy restricted-game analysis of the early-game long-term component candidates. The results of the analysis are striking (see Table 5.17). Using the PH07 early-game long-term procurement policy, C08-6 is the sole survivor of iterated elimination of dominated strategies, hence the sole rationalizable strategy in the restricted game. It is particularly interesting because of the simplicity of the policy and the large effect it imposes ($2M in this restricted game alone), even after the game designers' efforts to reduce a similar effect early in the competition's history (see Wellman et al., 2005a).

Candidate	NE Regret	Max Regret
C08-0	1.85	3.07
C08-5	1.93	3.76
C08-6	0	0
C08-7	1.07	2.52

Table 5.17: Comparison of 2008 candidate long-term procurement policies using stability metrics ($M).

After the early-game long-term procurement experiments, we introduced candidates C08-8 through C08-16. Candidates C08-8 and C08-9 modified the *bid improvement* policy of DeepMaize using a simulated annealing and genetic-algorithm based method, respectively. The candidates met with limited success and are not improving deviations to the C08-06 equilibrium. Experiments involving candidates C08-{10,14,15} once again focused on prediction methods, this time complemented with the PH07 procurement policy. Candidate C08-14 is an extension of C08-6 that included a bug fix in the component price predictor. Candidates C08-10 and C08-15 add an additional customer dataset and use the component price predictor of C08-2, respectively. Candidate C08-10 is not a positive deviation from C08-06, while C08-14 and C08-15 has small (less than 1%) and large (40%) equilibrium support in their respective 2-strategy reduce-games involving C08-6. Even though a bug was found in the interpolation method, both C08-6 and C08-15 outperformed C08-14 in strategic analysis. Reasons for this are unclear, though some evidence suggests that the long-horizon prediction models actually performed worse than short-horizon prediction models *on long horizon predictions*. This may be due to incorrect REPTree depth or lack of attribute selection (thus likely overfitting) on the long-horizon models. This tendency could account for both why the INTRP-bug method and the AIO method outperformed the INTERP method, especially since INTRP-bug grants stronger weights to the shorter horizon model when interpolating. Candidates C08-6 and C08-15 are qualitatively similar in our 4-strategy restricted-game analysis and we chose to create variants of C08-6 for further development simply because we believed that we were more likely to improve the component predictions with the framework defined in C08-6.

After C08-6 was selected, we decided to explore the PH07 early-game long-term procurement strategy in greater detail, as the previous experiments suggested early-game procurement is the feature that has the largest effect on an agent's performance. Table 5.18 gives the settings used to define the candidates early-game procurement strategy. In 2-strategy restricted-game analysis, C08-6 dominates the C08-19 strategy, however C08-20 fares better against C08-6. All-C08-6 and all-C08-20 profiles are pure-strategy Nash equilibria (hence primitive formations); however, the max regret of C08-20 is 27% of the max regret of C08-6 and C08-20 has a higher payoff in its respective primitive formation. From the 2-strategy analysis, we added in the PH07 and TT07F strategies. Once again, C08-20 has a max regret that is 19% of the max regret of C08-6 and the only primitive formations contain C08-06 and C08-20, respectively. Replicator dynamics returns a mixed strategy with strong support (98.6%) from C08-20 and weak support (1.4%) from C08-6. At this point, C08-20 became the focal point of our analysis.

Candidate	Mid-term Orders	Mid-term Quantity/Day	Long-term Quantity/Day
C08-6		N/A	25
C08-19	✓	50	50
C08-20	✓	35	30
C08-21	✓	35	37
C08-23	✓	30	25
C08-24	✓	40	30
C08-25	✓	30	40

Table 5.18: Candidate strategy early-game procurement settings

Candidates C08-{21,22,23,24,25} all slightly perturb the C08-20 mid-term and long-term quantities. We used a 2-strategy analysis to assess the candidates. In all cases, C08-20 had support in the observed equilibrium (calculated through replicator dynamics) of at least 97%. In addition, C08-20 had a smaller max regret than each opposing strategy in the respective 2-strategy restricted game. As a sanity check, we tested the 5-strategy restricted-game of {C08-20, C08-24, C08-25, PH07, TT07F}. Both PH07 and TT07F are pruned through iterated elimination of dominated strategies. Further still, the only primitive formation consists of C08-20 alone.

The final candidate, C08-28, is a variant of the C08-20 strategy that uses capacity-based component price predictions. Given the impending tournament finals, we had time

Candidate	NE Regret	Max Regret
C08-6	0.70	3.44
C08-20	0	2.37
C08-24	0.40	2.41
C08-25	0.67	2.73
C08-28	0.52	1.46

Table 5.19: Large 2008 candidate comparison using stability metrics ($M).

for one last large test. We tested a 7-strategy restricted game consisting of {C08-6, C08-20, C08-24, C08-25, PH07, TT07F}. Again, TT07F is removed through iterated elimination of dominated strategies. All of the candidate strategies have support in the equilibrium returned by a sample run of replicator dynamics; however, the max regret of the candidates differs substantially, as shown in Table 5.19. Our final decision before the last round of the tournament was between C08-20 and C08-28. Both seemed strategically strong, however C08-20 had been thoroughly tested and its log files heavily scrutinized, whereas C08-28 was a relatively late-breaking strategy. In the end, we decided to use C08-20 as the DeepMaize 2008 finals agent. The primitive formation analysis was developed after the tournament. In this case, had the analysis been available it would have strengthened our conviction to play C08-20. The primitive formation analysis yields a single primitive formation that solely contains C08-20. The full development tree is shown in Figure 5.9.

The agents that competed in the 2008 TAC/SCM finals are listed in Table 5.20. Three of the six finals teams had agents in the 2007 TAC/SCM finals. DeepMaize had a strong performance, finishing first in 14 of the 18 games and never placing lower than second in any game. The winning average score $5.318M, beat the next best score by $3.085M. Based on our previous analysis of long-term procurement, I expect that the difference in average scores is largely due to equilibrium settings of our early-game long-term procurement parameters. The equilibrium analysis exposed an NE regret of $1–2M for deviating from the PhantAgent 07 early-game procurement strategy, based off of a modified DeepMaize 07

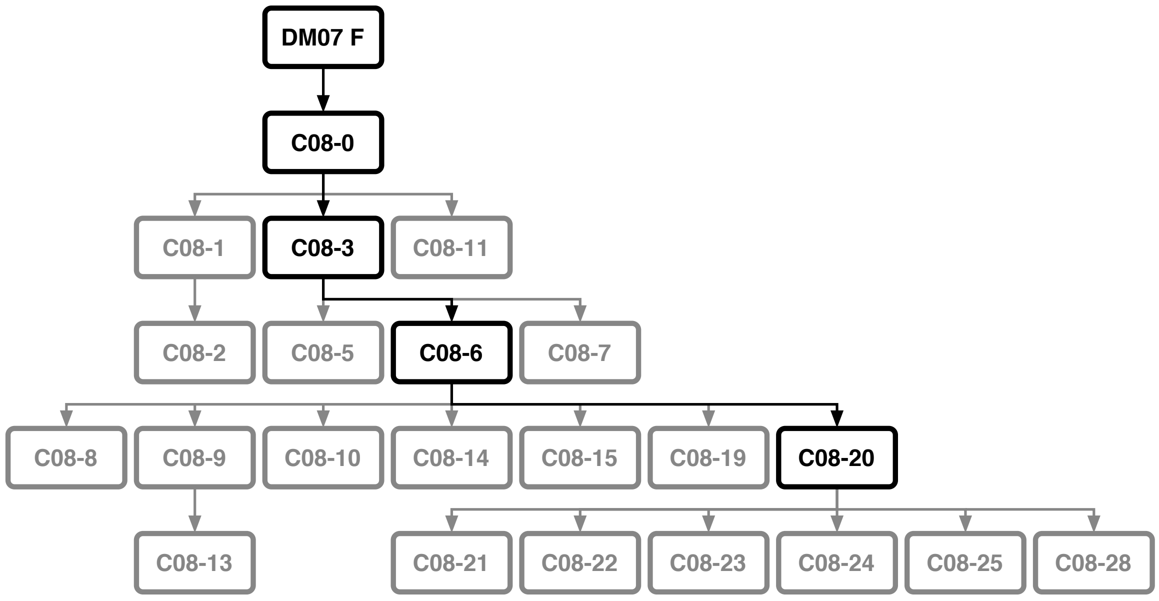


Figure 5.9: Full development tree for DeepMaize 08 with the chosen agent (C08-20) path highlighted. A feature matrix describing each candidate is given in Table 5.15.

Agent	Finals	Semifinals	Quarter-Finals	Seeding
DeepMaize	5.32	25.04 [A]	22.35 [A]	25.40
TacTex	2.23	8.072 [B]	16.50 [B]	24.83
CMieux	-0.84	4.912 [B]	17.23 [A]	19.82
CrocodileAgent	-5.38	4.490 [B]	16.04 [B]	16.49
Botticelli	-5.44	17.87 [A]	13.12 [A]	9.54
Merlion	-11.51	16.89 [A]	6.245 [A]	15.44

Table 5.20: TAC/SCM-08 finalists, with average scores ($M) from seeding through final rounds (semifinal and quarter-final groups in brackets). PhantAgent did not enter into the 2008 tournament.

strategy.

An unaltered version of DeepMaize 08 played and won the 2009 TAC/SCM tournament. However, the range of scores was smaller and the second highest-scoring agent (TacTex) lost to DeepMaize by only $0.34M on average. From conversations with the TacTex team, the 2009 agent differed only in a single parameter setting from the 2008 agent—the setting made TacTex less aggressive in long-term procurement and sales. This provides indirect evidence that the profile of agents in the TAC/SCM tournament finals is approaching equilibrium; however, a full empirical analysis (including all of the finals agents) is required to test that claim.

5.5 Discussion

The case study of the TAC/SCM market scenario illustrates the usefulness of the methods from previous chapters. First, through the use of the agent repository, I am able to compare successive years of agent strategies and provide evidence that agents are progressing. Agent progress is an important measure of the success and vitality of research competitions like TAC. Moreover, using the deviation and equilibrium analysis from Chapter 3, I determined low-regret strategies that had been previously regarded as weak by the tournament standings.

Second, I describe the development approach taken by the DeepMaize team spanning several competitions. This approach uses empirical game-theoretic analysis to evaluate and tune parameters in an agent. Through this analysis, I find several important aspects of agent behavior in TAC/SCM. In particular, I find that early-game procurement behavior remains one of the most critical aspects of agent design. In large part, this development approach led to championship finishes in the 2008 and 2009 tournaments.

Chapter 6
Searching for Approximate Equilibria

Given a simulation budget, which profiles should we simulate and how many observations should we take? A *profile selection policy* specifies a sequence of profiles to observe. Given an objective by which we can compare policies, determining the optimal policy is the *profile selection problem*. The methods I describe in this chapter for selecting profiles are motivated by the pivotal role stable profiles play in game-theoretic analysis and mechanism design. For these methods, the objective is to identify stable profiles.

With this in mind, I transform the *profile selection problem* into a *profile search problem* by letting each action in search correspond to a profile simulation and defining the goal of search as identifying an (approximate) equilibrium profile or set of profiles. The justification for framing the selection problem as a search problem is manifest in two observations. First, simulating is costly. For instance, simulation is the dominant cost in the TAC/SCM analysis of Chapter 5, taking nearly seven processor hours per data point. Second, simulating the entire game incurs an estimation cost in proportion to the size of the profile space, which is exponential in the number of players and the number of strategies available per player. If an (approximate) equilibrium can be identified without simulating the entire space, this can result in a substantial computational savings.

At least implicitly, when we select an approach to analyzing a strategic scenario, we have some set of objectives in mind. These objectives guide our choice of the model that is used for analysis as well as the methods we use in constructing the model. In this chapter, I consider two modeling objectives. The first objective is to identify a minimum-regret (pure-strategy) profile and the second is to obtain an accurate belief-distribution of minimum-regret (pure-strategy) profiles. This chapter focuses on pure-strategy profiles, whereas the policies in Chapter 7 consider mixed-strategy profiles. *Which objective is appropriate in what situation?* If, for example, the goal is simply to select a profile that, in expectation, is a low regret profile, then the first objective is appropriate. To illustrate the usefulness of the second objective, consider the NE regret of Section 3.1. The regret calculation uses a Nash equilibrium as a background context. The expected gain from deviation will vary according to which profile is chosen as the background context. Thus, we may want to evaluate a new strategy against a set of profiles that are believed to be low regret (approximate equilibria) and weight each score according to our belief that the respective profile is the minimum-regret profile. In such a situation, the intrinsic value to the modeler is having a robust estimate for regret. While both objectives are valid in some respective domain, the first objective has a quantitative metric that is easy to express. It is simply the expected regret of the profile selected at the end of the search process. It is this measure by which we compare prospective search algorithms.

I consider two distinct models of observation that influence the choice of a search process:

- *revealed-payoff model* – each observation determines the true payoff for a designated pure-strategy profile.
- *noisy-payoff model* – an observation draws a stochastic sample of the payoff for a designated pure-strategy profile

Algorithms for the revealed-payoff model are discussed in Section 6.2, and algorithms for the noisy-payoff model are discussed in Section 6.4. Common to both observational models, the outcome of a joint strategy is estimated by repeatedly simulating (sampling) the game. Previous research in the noisy-payoff model has explored directed sampling of profiles, by using value of information estimates (Walsh et al., 2003) or interleaving sampling and equilibrium calculations (Reeves et al., 2005). Both techniques require at least a small number of samples to be generated for every profile in the full joint strategy space.

Since it may be possible to establish that a particular profile is an equilibrium or near-equilibrium without considering all profiles, a search approach can potentially relax this requirement. This was part of the motivation for Sureka and Wurman (2005) proposing an algorithm based on tabu best-response search, an algorithm that searches for pure-strategy Nash equilibria within the profile space.

This latter algorithm is applicable in a *revealed-payoff* domain, where each search step determines the exact payoff for a designated pure-strategy profile. In contrast, the directed sampling methods described above assume a *noisy-payoff* model, where the basic search step corresponds to drawing a sample from an underlying distribution of payoffs.

I describe algorithms for both observational models, and compare them to the previous approaches from the literature. For the revealed-payoff model, the developed approach is based on a minimum-regret-first search. Additionally, I introduce a repeated sampling algorithm, termed information-gain search, applicable to the noisy-payoff model.

6.1 Notation

The algorithms used in this chapter assume a form of the empirical game model where $N^{\Gamma} = N^{\mathfrak{S}}$, $S^{\Gamma} = S^{\mathfrak{S}}$, and ϕ and μ are identity transformations on the profile and utility, respectively. It is assumed that u can be fit (parameters estimated) from the observation set Θ. Once u^{Γ} is fit, the empirical game $\mathcal{E}$ is fully specified. The notation $\Theta.s^k$ is used to indicate k more observations of profile s and $\mathcal{E}.s^k$ is the empirical game fit from those observations. Under the revealed-payoff model, each observation in Θ gives the value of the payoff function for some profile. Under noisy payoffs, each observation in Θ gives a noisy sample of the true payoff for some profile.

The search algorithms and analysis focus on *pure* profiles, where strategies are selected deterministically. Many but not all of the methods described can be extended in a straightforward manner to admit mixed strategies, where players choose actions probabilistically.

6.2 Search Methods for Revealed Payoffs

The revealed-payoff model involves games in which the simulator returns the true payoff (exact value of the estimated parameter) when queried with a particular profile. There are two search algorithms for this model studied previously in the literature: one based on TABU best response (or simply, TABU) by Sureka and Wurman (2005), and another applying regret bounds in a minimum-regret-first search (MRFS), employed by Vorobeychik et al. (2006).

TABU best response (Sureka and Wurman, 2005)

The TABU best-response algorithm begins by selecting an arbitrary profile as *active*. Subsequently, each iteration involves

(a) selecting a "deviant" player i,
(b) determining a best response, s_i^*, for i from the current active profile s,
(c) selecting the profile $s' = (s_i^*, s_{-i})$ as the next active profile,
(d) add an element to tabu list (L)—s_i in the *attribute based memory* version, or s in the *explicit memory* version of the algorithm.

When the attribute based memory version of the algorithm is used, the player i best-response strategy is selected from the remaining strategies not in the tabu list. When the explicit memory version of the algorithm is used, the player i best-response strategy is selected from the strategies not yielding profiles in the tabu list. The process terminates once the algorithm selects a PSNE as its active profile.

The original experiments by Sureka and Wurman evaluate performance based on the number of search steps required to find a PSNE. Since the experimenters know the base game and therefore its equilibria, they can simply terminate search when one of the known PSNE becomes the active profile.

In practice, when searching an unknown game, the algorithm cannot generally determine that an equilibrium profile is actually such when it first becomes active.[1] Thus, I also consider performance measures that require generated solutions to be *confirmed*—the profile and all of its deviations have been observed. I modify TABU to seek confirmation rather than move always to best response in order to address this requirement. In the modified version, instead of immediately placing the active profile on the tabu list L and branching to the best response, we do so only if the best response strictly increases the player's payoff; otherwise we keep the active profile unchanged. With this modification, we can confirm an active profile upon iterating through all the players. However, it now becomes possible in the explicit memory version of the algorithm (the version used in the experiments below) that we visit a profile for which all neighbors are in the tabu list. In this case we allow the player to deviate to the best response if that best response gives a higher payoff than the current profile. Pseudo-code for the tabu best-response algorithm used in the experiments is presented in Algorithm 5.

Algorithm 5 Tabu-Best-Response-Search

```
$T \leftarrow \emptyset$
Select initial profile at random
**while** *termination criteria not satisfied* **do**
    $i \leftarrow$ next player
    **if** $\mathcal{D}_i(s) \subseteq T$ **then**
        $s \leftarrow$ player $i$'s best response to $s$
    **else if** *$s$ has an improving deviation in $\mathcal{D}_i(s) \setminus T$* **then**
        Push $s$ onto $T$
        $s \leftarrow$ player $i$'s best response to $s$ not in $T$
```

[1]Moreover, in general we cannot assume that a PSNE even exists for the base game. We can relax the criterion to allow approximate equilibria, though we typically do not know *a priori* the regret of the best pure-strategy approximate solution.

Minimum regret first search

The idea of priority-first search is to expand a search tree by exploring the fringe node that is best according to some priority measure. In this setting, the objective is to find a profile minimizing the maximal gain from deviation. Therefore, I adopt as our priority measure a lower bound, $\hat{\epsilon}(s)$, on the possible gain to deviation from profile s, which is just the greatest gain among deviations from s that have been evaluated. The pseudo-code below describes this minimum-regret-first search (MRFS) procedure.

Algorithm 6 Minimum-Regret-First-Search

```
Select initial profile at random
while Queue is not empty do
    Select lowest $\hat{\epsilon}(s)$ profile $s$ from queue
    if $s$ is confirmed then
        Remove s from queue
        $\epsilon(s) \leftarrow \hat{\epsilon}(s)$
    else
        $\bar{s} \leftarrow$ Select-Deviation$(s)$
        Calculate $\hat{\epsilon}(\bar{s})$ from observations and insert $\bar{s}$ into queue
        Update $\hat{\epsilon}(\hat{s})$ for $\hat{s} \in \mathcal{D}(\bar{s})$ in the queue
```

The subroutine Select-Deviation(s) returns some deviation from s which has yet to be sampled. In the subroutine, we try to predict which unevaluated deviation from s is likely to give the largest gain from deviation. While the efficacy of the deviation selection heuristic depends on the game class, the following has worked well in a variety of cases. The heuristic tracks deviations by player and target strategy, and selects the unevaluated deviation from the current profile that has most frequently produced an improvement in the search history thus far.

6.3 Evaluating Search Methods for Revealed Payoffs

The experiments in this section employ games of various classes generated by GAMUT (Nudelman et al., 2005). When applicable, I select game instances similar in size to those used by Sureka and Wurman (2005). Initially I experiment with a game class used in their prior study to establish a baseline for algorithm comparison. I then proceed to investigate a game class whose structure is known to be exploited by a best-response dynamic, so that we may compare the algorithms in an environment expected to be favorable to TABU.

Uniform random games

The first class of games has payoffs that are uniformly and independently distributed in the range [-100,100]. The game class is denoted URG($|I|, |S_i|$), where $|I|$ is the number of players and $|S_i|$ is the strategy set size of each player. I compare MRFS and TABU on two sizes of games: the smaller URG(5,5) and the larger URG(5,10). To construct the data sets for comparing the algorithms, I generated 20 games in each class and checked which instances possess a PSNE. I ran each algorithm 100 times for each instance, with randomly selected starting profiles on each run.

The first comparison measures the number of evaluation steps (expressed in terms of percentage of profile space) required to confirm an equilibrium. For this measure, I necessarily limit attention to those games possessing a PSNE. The results of this comparison are shown in Table 6.1.

	URG(5,5)		URG(5,10)	
	Mean (%)	Median (%)	Mean (%)	Median (%)
MRFS	53.42	52.12	37.10	31.41
TABU	52.25	49.28	41.79	34.75
p value	0.18		3.5e-05	

Table 6.1: Percentage of profile space explored to confirm a PSNE, among URGs with at least one PSNE.

The analysis of URG(5,5) included 12 games which contained at least one PSNE. Seeds 0, 1, 3, 4, 8, and 15 contained one PSNE; seeds 5, 13, 16, 17, and 18 contained two; and seed 20 contained three. The performance of MRFS and TABU varied drastically according to the individual game. For instance, in seeds 4, 8, and 15, TABU rarely succeeded in confirming the solution.[2] Similarly in seed 3, MRFS on average requires nearly all the search space to be evaluated. It should be noted that although the equilibrium was not confirmed until near the last iteration, many near-equilibrium profiles were confirmed much earlier.

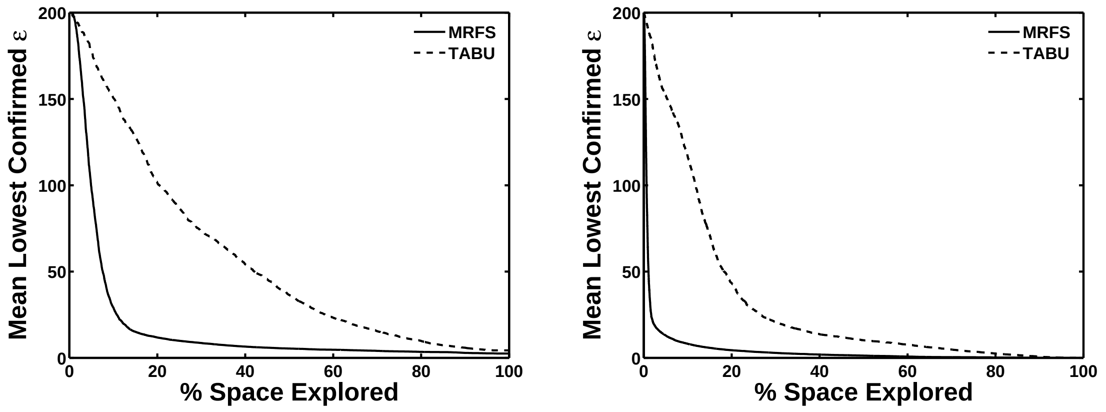


Figure 6.1: Mean lowest confirmed regret (ϵ) for URG(5,5) on the left and URG(5,10) on the right.

The analysis of URG(5,10) also included 12 games which contained at least one PSNE. Algorithm performance differences are statistically significant in the large game. In these larger games, the average performance of MRFS and TABU improves from approximately 50% of the space searched to the mid-30% range.

Figure 6.1 shows the minimum confirmed ϵ as a function of the space explored, which is our second performance measure. In many practical settings, it may be that near-equilibrium profiles are just as useful as PSNE. Therefore in those cases we consider the second measure more appropriate. Notice that MRFS confirms low ϵ profiles much earlier than TABU, which is the desired result.

[2]Since TABU is not guaranteed to confirm an existing solution, a timeout was placed on the number of iterations equal to the size of the strategy space. If TABU exceeded the timeout, it was credited for finding the solution in the greatest possible number of steps required by MRFS.

Congestion games

The second comparison I present is an experiment using *congestion games* (Rosenthal, 1973). In the congestion game generated by GAMUT (Nudelman et al., 2005), each player chooses a subset from the set of all facilities. Each player then receives a payoff that is the sum of payoff functions for each facility in the chosen subset. Each payoff function depends only on the number of other players who have chosen the facility. Congestion games exhibit two key properties for our purposes: they possess PSNE, and best-response learning processes converge to this equilibrium (Monderer and Shapley, 1996). This latter property suggests that the TABU best-response search algorithm should be effective.

I compare MRFS and TABU on four-player, four-facility congestion games. This class of games has 65,536 distinct profiles where each of the four players chooses a subset of the four facilities. Exploiting player symmetry reduces the size of the games to 3,876 distinct profiles, though in our experiments the algorithms do not exploit this symmetry. Exploiting symmetry would, of course, have only improved performance for the fixed game size. The results of the congestion game comparison are shown in Table 6.2. Twenty games were generated for experimentation using GAMUT. As expected, these games were extremely easy for TABU, which needed to search only a tenth of one percent of the profile space on average to confirm a PSNE. They were quite easy for MRFS as well, although this algorithm required 0.15 of one percent. The differences are statistically significant, but practically negligible given the miniscule amount of search required.

	MRFS	Tabu
Mean (%)	0.15	0.10
Median (%)	0.15	0.10
p value	$< 2.2\text{e-}16$	

Table 6.2: Mean percentage of profile space explored to confirm a PSNE in the four-player, four-facility congestion game class.

6.4 Search Methods for Noisy Payoffs

When payoff realizations are noisy, it stands to reason that the MRFS and TABU best-response algorithms will be inadequate, since these do not consider how to allocate samples across evaluated profiles. Nevertheless, we can apply them to the problem in a modified form, interpreting an evaluation step as a decision to draw k payoff samples for the target profile. Clearly, as k increases, so does reliability of the answers. On the other hand, increasing k reduces the number of profiles that can be explored with a given number of samples. An analyst can, perhaps, guess a reasonable value for k for a particular problem, using higher k when more noise is present. Such a solution is unsatisfactory for two reasons. First, it seems a waste to sample each profile an equal number of times, since some profiles are revealed to be unlikely solution candidates after only a few samples. Second, we would like to develop an algorithm that can automatically adjust sampling allocation appropriately for a given problem, rather than involve the analyst in the process.

I am aware of two previous approaches to the problem of sample allocation in games with noisy payoffs, one by Walsh et al. (2003) and another by Reeves et al. (2005). Walsh et al. introduced a search algorithm founded on the principles of metareasoning (Russell and Wefald, 1991), using an approximate regret function to determine the value of choosing a specific profile to sample. Reeves et al. proposed a method of search by which profiles are selected to be sampled according to the estimated probability mass placed on the specific profile in a sample equilibrium. I introduce another approach that is based on information gain as measured by the Kullback-Leibler divergence criterion (Kullback and Leibler, 1951).

EVI and ECVI (Walsh, Parkes, and Das)

First, I discuss the approach to guided search in noisy games introduced by Walsh et al. (2003). I use the symbol s to denote a profile in S as well as the action of sampling that profile and the symbol S^T to represent all sequences of sampling actions of length T. Let φ be some decision model that maps an empirical game $\mathcal{E}$ to a profile $s \in S$, denoted by $\varphi(\mathcal{E})$. Let ψ be the *error model* and $\psi(s|\mathcal{E})$ be the *error* for selecting a profile s in the game $\mathcal{E}$. Walsh et al. define the *expected value of information* (EVI) from sampling a particular profile s under the current information state $\mathcal{E}$ to be

$$\mathrm{EVI}(s|\mathcal{E}) = \mathbb{E}_{\mathcal{E}.s|\mathcal{E}} \left[\, \psi(\varphi(\mathcal{E})|\mathcal{E}.s) - \psi(\varphi(\mathcal{E}.s)|\mathcal{E}.s) \, \right]. \tag{6.1}$$

Walsh et al. propose two algorithms. The first of these is EVI as defined in Equation 6.1, in which $\varphi(\mathcal{E})$ selects an arbitrary, possibly mixed-strategy, Nash equilibrium in the empirical game $\mathcal{E}$. Additionally, their error model is *cumulative regret*, defined to be

$$\psi(s|\mathcal{E}) = \sum_{i \in N, \ \hat{s} \in \mathcal{D}_i(s)} \max(0, \ u_i^{\mathcal{E}}(\hat{s}) - u_i^{\mathcal{E}}(s)).$$

Finally, they develop a particular model of future information which uses distributional estimates for the payoffs $u_i(s)$.

Walsh et al. use Monte Carlo simulation to calculate the expectation of $\psi(\varphi(\mathcal{E}.s)|\mathcal{E}.s)$. For each sample of s, an additional set of Monte Carlo simulations estimate the decision process $\varphi(\mathcal{E}.s)$. Walsh et al. found that this EVI approach is computationally infeasible for large games, which motivated their development of a second algorithm termed *expected confirmational value of information* (ECVI). Whereas in EVI sampling s has positive value in expectation only if it is expected to change (refute) the current equilibrium choice, ECVI gives more value to s if it is likely to *confirm* the current equilibrium $\varphi(\mathcal{E})$. Specifically, the authors calculate ECVI as

$$\mathrm{ECVI}(s|\mathcal{E}) = \mathbb{E}_{\mathcal{E}.s|\mathcal{E}} \left[\, \psi(\varphi(\mathcal{E})|\mathcal{E}) - \psi(\varphi(\mathcal{E})|\mathcal{E}.s) \, \right].$$

To help understand the implications of using ECVI, consider an alternate method for calculating the expected error of $\psi(\varphi(\mathcal{E}.s)|\mathcal{E}.s)$. Instead of generating Monte Carlo samples for $\psi(\cdot)$ and $\varphi(\cdot)$ independently, we use each additional sample for the error and decision function's empirical game $\mathcal{E}.s$. Because a Nash equilibrium will always exist in our finite $\mathcal{E}.s$, we know that one will be selected by the decision process $\varphi(\mathcal{E}.s)$. We know from the definition of the error function that for each Monte Carlo sample, $\psi(\varphi(\mathcal{E}.s)|\mathcal{E}.s)$ will be zero. Therefore, the EVI equation simplifies to

$$\mathrm{EVI}(s|\mathcal{E}) = \mathbb{E}_{\mathcal{E}.s|\mathcal{E}}\left[\, \psi(\varphi(\mathcal{E})|\mathcal{E}.s)\right]. \tag{6.2}$$

Notice that in ECVI, the left term in the expectation is a constant independent of s. Both EVI and ECVI seek to maximize the value of their respective information measures. EVI chooses the s that maximizes the expectation in Equation 6.2, whereas ECVI chooses the s that minimizes it.

Intuitively, the left and right terms in the expectation of Equation 6.1 provide an implicit balance between exploitation and exploration, respectively. The exploration component vanishes in Equation 6.2. Notice that EVI chooses a profile s that is the most likely, in expectation, to refute the current candidate solution. This is precisely what MRFS attempts in the revealed-payoff domain by selecting unobserved deviations from the current candidate solution. Thus, Equation 6.2 also provides a qualitative link between EVI and MRFS.

Information gain approach

Approaches such as EVI and ECVI focus on improving a particular (perhaps arbitrary) Nash equilibrium estimate (in our setting, a pure strategy equilibrium). The information gain approach focuses on improving an estimate that is based on any model that yields a *distribution* over profiles given an empirical game. Such profile distributions arise, for example, as *belief distributions of play* (Vorobeychik and Wellman, 2006), which are beliefs constructed by an outside observer (e.g., mechanism designer) about the relative likelihood of different profiles arising as a result of actual strategic interaction that is modeled by the game. Belief distributions of play may model players as selecting an arbitrary Nash equilibrium, or may involve more complex beliefs—for example, a probability distribution which assigns higher probability to profiles with lower regret will likely assign positive (albeit often small) probability to every profile in a finite game (Duong et al., 2008). We are interested in a particular such belief model that assigns a relative likelihood to profiles based on their respective probabilities of having the smallest regret.

The information gain algorithm is, in principle, straightforward. We begin by presuming that our sampling action will take k samples. With each profile $s \in S$ we compute (or approximate) information gain from sampling this profile k times. We then select the profile that promises the greatest information gain. The core of the approach to computing information gain, based on Kullback-Leibler divergence, is very general in that it can use any prior distribution on profiles obtained based on the current empirical game, $p_s(\mathcal{E})$. Thus, I first develop it for an arbitrary distribution, and then specialize to one of particular interest in this work.

First, I define the *entropy* of a profile s, $\mathcal{H}(s;\mathcal{E})$:

$$\mathcal{H}(s;\mathcal{E}) = -p_s(\mathcal{E})\log_2 p_s(\mathcal{E}) - (1-p_s(\mathcal{E}))\log_2(1-p_s(\mathcal{E})).$$

The standard definition of *cross entropy* of s, denoted here by $\mathcal{H}(s;\mathcal{E},\hat{\mathcal{E}})$, is then

$$\mathcal{H}(s;\mathcal{E},\hat{\mathcal{E}}) = -p_s(\mathcal{E})\log_2 p_s(\hat{\mathcal{E}}) - (1-p_s(\mathcal{E}))\log_2(1-p_s(\hat{\mathcal{E}})).$$

Based on these, I define the *information gain* for a profile s from taking k additional samples of $\hat{s}$, denoted $\mathcal{G}(s;\mathcal{E},\mathcal{E}.\hat{s}^k)$, to be

$$\mathcal{G}(s;\mathcal{E},\mathcal{E}.\hat{s}^k) = \mathcal{H}(s;\mathcal{E},\mathcal{E}.\hat{s}^k) - \mathcal{H}(s;\mathcal{E}).$$

We determine the profile to sample by comparing the aggregate information gain over all profiles from taking k additional samples of a profile. However, additional samples of profile $\hat{s}$ update the posterior distribution of a profile s, $p_s(\mathcal{E}.\hat{s})$, only if $\hat{s} \in \mathcal{D}(s)$. Therefore, the aggregate information gain from sampling a profile $\hat{s}$ a total of k times, denoted $\mathcal{G}(\mathcal{E}, \mathcal{E}.\hat{s}^k)$, is computed as

$$\mathcal{G}(\mathcal{E}, \mathcal{E}.\hat{s}^k) = \sum_{s \in D(\hat{s})} \mathcal{G}(s; \mathcal{E}, \mathcal{E}.\hat{s}^k).$$

The information gain so defined is then used as a part of our INFO-GAIN-SEARCH selection algorithm:

Algorithm 7 INFO-GAIN-SEARCH($\mathcal{E}(\emptyset), k, T$)

```
Select initial profile at random $s$
$\mathcal{E} \leftarrow k$ samples of $s$
while Termination criteria not satisfied do
    $s \leftarrow \arg\max_{\hat{s}} \mathbb{E}_{\mathcal{E}.\hat{s}^k|\mathcal{E}} \left[\mathcal{G}(\mathcal{E}, \mathcal{E}.\hat{s}^k)\right]$
    $\mathcal{E} \leftarrow \mathcal{E} \cup k$ samples of $s$
return $\arg\max_{\hat{s}} \; p_{\hat{s}}(\mathcal{E})$
```

In developing the assessment of likely strategic outcomes based on the evidence encompassed by the empirical game, I posit that players are most likely to play a profile with the lowest regret. Since we restrict our search space to pure strategy profiles, such profiles need not constitute Nash equilibria, although often they will (particularly in very large games), and even more often the smallest regret will be indeed quite low to justify our belief. Thus, I define the information gain with respect to the distribution $p_s(\mathcal{E})$ that assigns probabilities to profiles s in proportion to their likelihood of having the smallest regret. I now develop these distributions formally, beginning with the definition of the highest payoff a player i can obtain by deviating from s to another strategic option.

Definition 25 (Maximum deviation payoff)**.** *For a given player i and profile s, the maxi-*

mum deviation payoff is

$$g_i(s) = \max_{\hat{s}_i \in S_i \setminus s_i} u_i(\hat{s}_i, s_{-i}).$$

The distribution of $g_i(s)$, denoted by $F_{g_i(s)}(d)$, is the n^{th} order statistic (maximum) over the mean payoffs of the deviations, given by

$$F_{g_i(s)}(d) = \prod_{\hat{s}_i \in S_i \setminus s_i} F_{u_i(\hat{s}_i, s_{-i})}(d).$$

The distribution of player regret, r, denoted by $F_{\delta_i(s)}(r)$ can be obtained by conditioning on the payoff to i from playing s:

$$F_{\delta_i(s)}(r) = \int_{\mathbb{R}} F_{g_i(s)}(u + r)\, dF_{u_i(s)}(u). \tag{6.3}$$

We estimate the integral in (6.3) using a Monte Carlo method with importance sampling (see Chapter 2). The distribution of regret for a particular profile, s, is then simply

$$F_{\epsilon(s)}(r) = \prod_{i \in I} F_{\delta_i(s)}(r).$$

Now, as the final piece, we can define the actual distribution of minimum regret, that is, we can define, for each profile $s \in S$, the probability that s has minimum regret given the evidence in the empirical game:

$$p_s(\mathcal{E}) = \int_{\mathbb{R}} \left[\prod_{\hat{s} \in S \setminus s} \left(1 - F_{\epsilon(\hat{s})}(r)\right) \right] dF_{\epsilon(s)}(r). \tag{6.4}$$

To estimate the value of the integral in (6.4) using Monte Carlo simulation, we have to generate M realizations of the random variable. Each of these M realizations requires computing or estimating the integral in (6.3) a total number of $|S| - 1$ times. The latter, as I already mentioned, is also estimated using Monte Carlo methods by generating N realizations of its respective random variable. Thus, each iteration requires $\mathcal{O}(|S|NM)$ operations, for at total running time of $\mathcal{O}(|S|NM\frac{T}{k})$. Furthermore, we may have to use a very large M to get a reasonable approximation. Consequently, running time of our algorithm quickly becomes impractically long. To keep it somewhat in check, I instead approximate the integral by using point estimates for the mean regret of the remaining profiles:

$$p_s^*(\theta) = F_{\epsilon(s)}(\epsilon^{(1)}_{S\setminus s}), \tag{6.5}$$

where $\epsilon^{(1)}_{S\setminus s}$ is the lowest regret over all profiles except s calculated using the expected mean payoffs given the empirical game $\mathcal{E}$. The approximation in Equation 6.5 requires $\mathcal{O}(|S|N)$ calculations in each iteration, for a total running time of $\mathcal{O}(|S|N\frac{T}{k})$.

6.5 Evaluation of Search Methods for Noisy Payoffs

Unlike the evaluation of search algorithms for games with revealed payoffs, which used randomly generated games of various classes, I evaluate the approaches for noisy games in a more representative setting—the TAC/SCM game described in Chapter 5. The scenario is modeled as a symmetric normal form game with five heuristic strategies,[3] for a total of 35 strategy profiles. The payoffs in the game are estimates based on the analysis of Section 5.3. I use the sample mean payoffs to construct a base game that has structure similar to the empirical TAC/SCM model. Therefore, it is with respect to the approximated TAC/SCM model that we measure error. This technique, in our application, creates three player-pairs. Each of these pairs is constrained to play the same strategy and the payoff to the pair is the average of the payoffs of each member in the base game. In this study, I add zero-mean Gaussian noise on top of the already sampled base-game mean payoffs to mimic the simulation process.

I present the results of two experiments. The first experiment uses Gaussian noise with

[3]The heuristic strategies are a subset of the agents who participated in the TAC/SCM 2006 tournament and released binary versions of their agent software.

standard deviation of 3.75 million, which is roughly the order of magnitude of the noise found in TAC/SCM simulations. The second experiment has a larger standard deviation of 10 million. For each of these experiments, we tested four different algorithms. For each experiment and algorithm, I generated 100 runs and average the score over runs. The score is the true regret ϵ of the returned profile in the base game as a function of the number of samples.

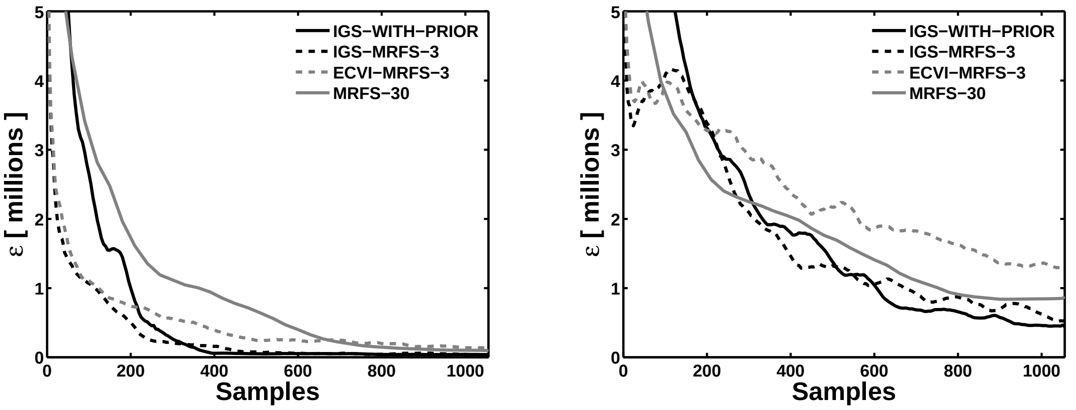


Figure 6.2: Mean regret (ϵ) in the SCM$\downarrow_3$ 2006 game of the profile returned by the algorithms after a given number of samples when small variance (left) and large variance (right) Gaussian noise is added to the payoffs.

The first algorithm tested was the MRFS extension to noisy games, labeled MRFS-30 in Figure 6.2. MRFS-30 samples each profile 30 times and uses the resultant mean as if it were the actual payoff in a revealed payoff game.

Second, I tested the IGS and ECVI algorithms. These are repeated sampling algorithms and normally require some initial samples of every profile in the game. Therefore, I prefaced the repeated sampling portion of the search with an MRFS search, where three samples are taken per profile. Each iteration of IGS and ECVI algorithms took five samples of each profile per iteration. These algorithms were labeled IGS-MRFS-3 and ECVI-MRFS-3, respectively.

Finally, I tested the IGS algorithm with a zero-mean Gaussian prior payoff distribution over profiles. In the small-variance game, the standard deviation of the Gaussian prior was taken to be five million, whereas in the large-variance game it was 20 million. This algorithm was labeled IGS-WITH-PRIOR.

Figure 6.2 shows the results of the analysis. In the small variance game, we see that MRFS-30 does not perform as well as the other algorithms for most of the sample sizes. Using MRFS-3 to gather initial samples seemed to help substantially for the first 200 samples, after which IGS-WITH-PRIOR caught up with the performance of IGS-MRFS-3. Note that MRFS-3 uses up the first 105 samples. Finally, I note that IGS-MRFS-3 offers a performance improvement over ECVI-MRFS-3.

In the large-variance game we note the surprising result that all of the algorithms outperformed ECVI-MRFS-3, particularly when more samples were taken. MRFS-30 displays a particularly strong performance in this game class, essentially on par with IGS-MRFS-3 and IGS-WITH-PRIOR.

6.6 Discussion

I have investigated the problem of searching for approximate equilibria in games where determining the payoff for particular profiles is costly. For each observation model, I experimentally evaluated the known approaches from prior literature—all, as far as I am aware—along with new algorithms and variants, on a range of game instances and game classes.

For the revealed-payoff model, I compared MRFS and TABU on classes of games with or without helpful structure. In all cases, I found that the methods require approximately the same number of search steps on average to confirm a PSNE. MRFS significantly outperforms TABU, however, in terms of its ability to confirm better approximate equilibria earlier, for games that require significant search. Another important attribute of the MRFS algorithm is that it will confirm all available profiles eventually, whereas TABU may not. This is important not only in the case where no PSNE exists, but also when we wish to analyze low-regret profiles when designing a best response.

For the noisy-payoff model, I introduced a new algorithm based on information gain, called IGS, and found that it outperforms the ECVI repeated sampling algorithm, the current benchmark in the literature. Unlike ECVI, which was an attempt to construct a computationally feasible version of expected value of information, the IGS family of algorithms does not directly resolve to improve the mean estimate of player regret. Instead, the IGS algorithm focuses on improving some distribution over profiles. For example, this distribution could be a distribution of play, a likelihood of PSNE, or the probability that a profile minimizes regret. Optimizing the distribution rather than a point estimate can improve calculations involving the distribution and other heretofore unknown quantities. Consequently, an EVI-based algorithm may not completely capture the decision theoretic-problem underlying the game analysis task.

In addition, IGS does not succumb to a problem that plagues ECVI. That is, ECVI has a tendency to sample *safe* profiles, or precisely, profiles that are not likely to change the current decision in expectation. Thus, ECVI can easily get stuck in local optima and never recover.

Although MRFS was developed and justified under the revealed-payoff model, I have shown that even under noisy payoffs, using MRFS to select initial samples can improve performance early on in the search process. Moreover, I have shown that in some cases MRFS can perform as well as IGS when sampling noisy games.

One significant drawback to MRFS is the constant number of samples per iteration. Given the relative strength MRFS has displayed on the tested classes of games, an interesting future path of study is a dynamic variant of MRFS which takes into account the significance of the deviation comparisons to determine how many times to sample a profile.

Chapter 7
Searching for Approximate Formations

In this chapter, I consider a special case of the profile selection problem: policies that specify a sequence of restricted games to evaluate. I define such policies as *restricted-game selection policies*. Because evaluating a restricted game involves evaluating all of its constituent profiles, a restricted-game selection policy implicitly specifies a profile selection policy. Given an objective measure of quality, determining an optimal restricted-game selection policy is the *restricted-game selection problem*, which is also termed the *strategy exploration problem* in the existing literature.

In existing literature, policies are measured by the regret of a profile selected after the execution of the policy—the same as in Chapter 6. However, because the policies produce restricted games that are completely observed, we can determine the regret of the resultant restricted game—the largest regret, induced by strategies not in the restricted game, of any profile in the restricted game. In effect, policies are measured according to their ability to *minimize the minimum regret* with respect to the base game (minimize the regret of a minimum-regret profile) and *minimize the maximum regret* of all the profiles in the restricted game with respect to the strategies not contained in the restricted game (minimize the formation regret of the restricted set of strategies).

I propose a novel policy that seeks to identify (approximate) formations—restricted games that are minimal according to the second measure. This formation-based policy performs as well as or better than the existing policies on all measures in the experiments.

7.1 Strategy Exploration Problem

Schvartzman and Wellman (2009a) were the first to address the strategy exploration problem. The authors observed that, in practice and existing literature, empirical game-theoretic analysis often involves dynamic formulation of a game model: we evaluate *a priori* salient strategies—for instance, winners in a TAC tournament—then, based on intermediate analysis, we evaluate additional candidates. This approach is consistent with the profile selection policies of Chapter 6, except that the intermediate analysis makes use of fully-observed restricted games. For instance, the empirical analysis of TAC/SCM in Chapter 5 that led to the design of DeepMaize, involved repeated evaluation of small restricted games. In large part, this is because restricted games allow for easy mixed-profile analysis, especially when the support consists of only a few strategies. Profile selection policies such as EVI and IGS also allow for analysis and identification of mixed-profiles, however these algorithms require a small number of observations for all profiles, which is often infeasible due to computational constraints. Thus, Schvartzman and Wellman investigate policies that focus on restricted-game evaluation, where the objective is to identify a minimum-regret profile. I consider an additional objective: *identify an approximate formation*.

A policy determines a *sequence of restricted games* $\Gamma_{S\downarrow X^{(0)}}, \ldots, \Gamma_{S\downarrow X^{(k)}}$ to be evaluated, where $X^{(0)} \subset \cdots \subset X^{(k)} \subseteq S$. Each restricted game $\Gamma_{S\downarrow X^{(j+1)}}$ is formed by making additional strategies available to players in $\Gamma_{S\downarrow X^{(j)}}$. I consider policies for the revealed-payoff model (Section 6.2), in which each observation determines the true payoff for a designated pure-strategy profile. In this case, *evaluating a restricted game* means observing the payoffs for each pure-strategy profile in $X^{(j)}$.

Schvartzman and Wellman (2009a) proposed four basic policies, where the policies add a single strategy to X during each step. The policies are described using symmetric games—therefore S and X refer to strategy sets rather than sets of profiles—however, they can be easily extended to non-symmetric games. The first policy, random (RND), picks one of the remaining strategies in $S \setminus X$ with equal probability. The remaining policies choose a strategy from the *improving deviations*. Strategy $s'_i \in S_i$ is an *improving deviation* for player i with respect to profile σ, if i would benefit by playing s'_i rather than its designated strategy in σ: $u_i(s'_i, \sigma_{-i}) > u_i(\sigma)$. Let $\mathfrak{D}_i(\sigma)$ be the set of improving deviations with respect to σ for player i and $\mathfrak{D}(\sigma) = \cup_{i \in N} \mathfrak{D}_i(\sigma)$—because of symmetry $\mathfrak{D}_i(\sigma) = \mathfrak{D}_j(\sigma)$ for all $i, j \in N$. The following deviation policies are described by Schvartzman and Wellman:

- Improving deviations only (DEV): Find a Nash equilibrium, σ, of the current restricted game, and choose a strategy uniformly from $\mathfrak{D}(\sigma) \setminus X$.
- Best response (BR): Find a Nash equilibrium, σ, of the current restricted game, and choose a strategy uniformly from $\mathcal{B}(\sigma) \setminus X$.
- Softmax (ST): Find a Nash equilibrium σ of the current restricted game. Choose strategy s'_i from $\mathfrak{D}(\sigma) \setminus X$ with probability given by the softmax formula applied to deviation gains with temperature parameter T and scaling parameter α: $\alpha e^{(u(s'_i, \sigma_{-i}) - u(\sigma))/T}$.

The authors evaluate the k-step strategy exploration policies by measuring the regret with respect to Γ of a sample Nash equilibrium of $\Gamma_{S \downarrow X^{(k)}}$. They find that improving deviation policies perform better than unrestricted selection policies—such as RND.

With the exception of RND, the policies proposed by Schvartzman and Wellman require knowledge of the payoffs for single-player deviations from profiles in $X^{(j)}$ to the remaining strategies.[1] In other words, the utility functions for each player i is defined over $S_i \times X^{(j)}_{-i}$ for the jth step in the exploration. I define a new concept that encapsulates this model of a game, called an *augmented restricted game*.

Definition 26 (Augmented Restricted Game). *Let* $\Gamma^{*}_{S \downarrow X}$ *be an* augmented restricted game *with respect to the* base game Γ, *where players in* $\Gamma^{*}_{S \downarrow X}$ *are restricted to playing profiles in* $X \subseteq S$, *however the utility function for each player* i *is a mapping* $u_i : S_i \times X_{-i} \to \mathbb{R}$.

From Section 4.3, we can calculate the base-game regret of any profile in $\Gamma_{S \downarrow X}$ by calculating its restricted-game regret, if X is a formation. In addition, our estimate can be understating the base-game regret by no more than ϵ, if X is an ϵ-formation. Contrast

[1]In the worst case, this may require all single-player deviations to each of the remaining strategies. Exceptions to this requirement may occur, for instance, if the policy has access to an *oracle* that selects the best response out of the remaining strategies.

this with calculations that can be performed with augmented restricted games: given the augmented restricted game $\Gamma^{*}_{S\downarrow X}$, we can calculate the base-game regret for *any* profile in $\Gamma_{S\downarrow X}$, regardless of X's status as a formation; furthermore, given $\Gamma^{*}_{S\downarrow X}$, we can calculate the regret of X (see Definition 23) without any additional profile observations. Therefore, we now state the strategy exploration problem as determining a sequence of *augmented restricted games* $\Gamma^{*}_{S\downarrow X^{(0)}}, \ldots, \Gamma^{*}_{S\downarrow X^{(k)}}$ to be simulated, where $X^{(0)} \subset \cdots \subset X^{(k)} \subseteq S$.

Schvartzman and Wellman propose that we evaluate a policy according to the regret of a profile (Nash equilibrium) produced at each iteration. I propose an additional metric, where we evaluate a policy according to the regret of the restricted strategy set (the joint strategy set of the restricted game) produced at each iteration. These metrics correspond to two different modeling objectives: *minimize the minimum regret* and *minimize the maximum regret*.

Evaluating the two metrics introduces interesting computational problems. To minimize the maximum regret, I modify the FIND-FORMATION algorithm (see Algorithm 4 of Section 4.6), such that the set X—the restricted strategy set that results from the policy—is the bound rather than the size of the resultant restricted game. Finding a (mixed) profile with minimum regret is also an interesting new problem. Schvartzman and Wellman selected a profile using replicator dynamics on the restricted game. In many cases replicator dynamics finds a Nash equilibrium, however equilibria in the restricted game are not necessarily minimum-regret profiles—when constrained to mixtures over X—in the base game. In the following section, I describe an algorithm for finding minimum-regret profiles when we are constrained to mixtures over X.

7.2 Minimum-Regret Constrained Profiles

Consider the augmented reduced game, $\Gamma^{*}_{S\downarrow X}$, that results from some policy. We may wish to determine a minimum-regret profile of Γ, using our limited knowledge of the payoffs in $\Gamma^{*}_{S\downarrow X}$—for instance, identifying a minimum-regret profile allows us to measure the minimum profit-regret of a policy (the first evaluation metric). This minimum-regret profile, σ, is *constrained* by the domain of the utility functions, such that $\sigma \in \Delta^X$, where $\Delta^X = \times_{i\in N}\Delta(X_i)$. A minimum-regret profile of Γ is a Nash equilibrium, therefore we may be tempted to select a Nash equilibrium of $\Gamma_{S\downarrow X}$. However, a *minimum-regret constrained profile* is not an equilibrium of the restricted game in general. We can, however, identify a minimum-regret constrained profile (MRCP) by solving the following optimization problem:

$$\arg\min_{\sigma\in\Delta^X} \epsilon(\sigma),$$

where $\epsilon(\cdot)$ is with respect to Γ. This is a constrained optimization problem with a nonlinear, non-differentiable objective function and both inequality and equality constraints. Because the regret function is non-differentiable, standard optimization techniques that calculate the gradient of the Lagrangian do not apply. Various *direct search algorithms* have been proposed to solve optimization problems where a gradient is not available or efficiently calculable. Powell (1998) and Kolda et al. (2003) provide a good review of these algorithms. In practice, one of the most popular algorithms is the *amoeba method*, introduced by Nelder and Mead (1965), that iteratively refines a simplex in the search space until convergence or some fail condition is reached. In a similar setting, Walsh et al. (2002) used the amoeba method to calculate Nash equilibria of an empirical game representing a continuous double auction scenario.

When applying the amoeba method to the MRCP optimization problem, we have to reconcile the fact that the optimization problem is constrained and the amoeba method is an unconstrained optimization technique. One way to account for this is to optimize over a different objective function, where the new function equals $\epsilon(\sigma)$, if $\sigma \in \Delta^{\Gamma^{*}_{S\downarrow X}}$, and infinity, otherwise. The problem with this approach is that the amoeba algorithm searches over $\mathbb{R}^{|X|}$ and the feasible region of the space has zero measure with respect to $\mathbb{R}^{|X|}$.

To handle the constraints, we have a few general approaches: apply a penalty function, transform the variables, or restrict the amoeba method's vertex generation steps to feasible regions. Penalty functions are problem dependent and care must be taken not to introduce new local minima with their application. We can transform variables by introducing $x(s_i)$ where $\sigma(s_i) = x^2(s_i)$ for every $s_i \in X_i$; however, this relieves the inequality constraints, but not the equality constraint.

What about maintaining feasibility? If an iteration starts with a simplex where each vertex is within the feasible region, then we can modify the amoeba method such that we always end the iteration with a feasible simplex. To do this, we modify the reflect and expand steps of the original amoeba method to generate only feasible vertices. We use a binary search to select the maximum feasible reflection (α) and expansion (γ) scaling parameters, respectively, if the unmodified reflected or expanded vertex is infeasible. Using this approach, all the vertices of the simplex are feasible at the end of each iteration.

I now make an additional observation. Minimum regret profiles may not have full support. In fact, minimum regret profiles will often have small support, a feature that is exploited by equilibrium finding algorithms like those introduced by Porter et al. (2008). If the minimum regret profile does not have full support, then the inequality constraints on σ are active at the minimum.

Figure 7.1 shows a sample run of the modified amoeba method on a three-strategy restricted TAC/Travel game, overlaid on the regret contour. The method starts by generating feasible vertices uniformly over the probability simplex consisting of the three strategies. The simplex quickly reduces support from strategy 19. However, the second time the simplex meets the strategy 19 boundary, it collapses into a simplex with almost zero measure. Once this occurs, the amoeba algorithm does recover. Fortuitously, the amoeba algorithm converges near the local minimum in this example, however this will not be the case in general.

One way to deal with this problem is to search over small supports. Notice that for $\hat{X} \subseteq X$, the MRCP in $\hat{X}$ is a candidate for the MRCP in X. To find the minimum re-

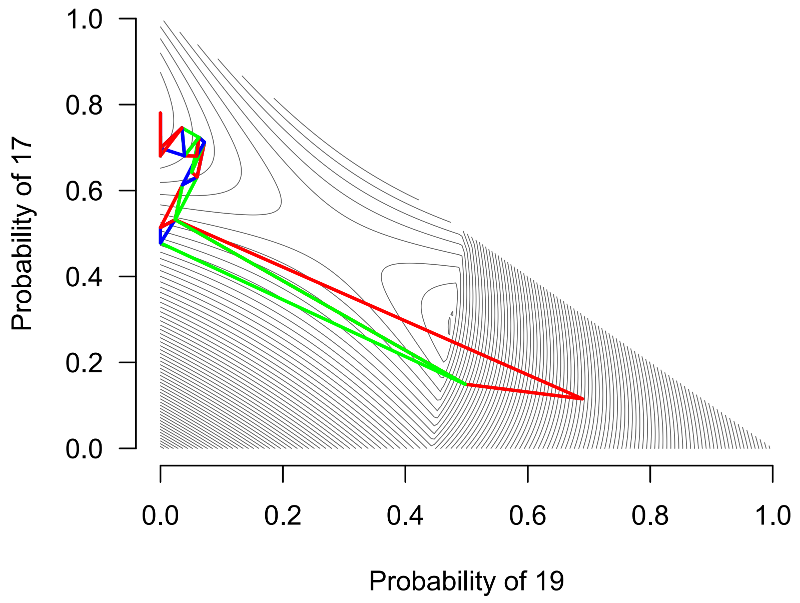


Figure 7.1: The sample run of the amoeba method on TAC/Travel.

gret profile in $\Gamma^{*}_{S\downarrow X}$, we can search over subsets of X. All else being equal, we are more likely to converge to the minimum when it is an interior point of $\Delta(\hat{X})$, than a boundary point of $\Delta(X)$. Algorithm 8 gives a recursive algorithm for solving the MRCP optimization problem, where AMOEBA-MRCP is the modified amoeba algorithm for finding the MRCP.

Algorithm 8 FIND-MRCP($\Gamma^{*}_{S\downarrow X}$)

```
$\Sigma \leftarrow \emptyset$
**for** $\hat{X} \subset X$ *where* $\hat{X}$ *has a single strategy removed from* $X$ **do**
    $\Sigma \leftarrow \Sigma \cup$ FIND-MRCP($\Gamma^{*}_{S\downarrow \hat{X}}$)
$\Sigma \leftarrow \Sigma \cup$ AMOEBA-MRCP($\Gamma^{*}_{S\downarrow X}$)
**return** $\arg\max_{\sigma \in \Sigma} \epsilon(\sigma)$
```

Notice that we repeatedly compute the MRCP for small restricted strategy sets. Using memoization we can dramatically improve performance. Moreover, strategy exploration policies iteratively add strategies to the existing strategy set. Therefore, we can also retain the computed values for later calls to FIND-MRCP.

7.3 Formation Policies

In this section, I propose a novel formation-based policy that is closely related to the improving-deviation policies investigated by Schvartzman and Wellman (2009a). Consider the best response (BR) policy: at the k^{th} iteration, pick a Nash equilibrium of $\Gamma_{S\downarrow X^{(k)}}$; choose a best-response strategy from among the remaining strategies; and use that strategy to construct $X^{(k+1)}$. The basic idea behind BR is successively refuting the current minimum-regret profile. This is a common theme among profile-search algorithms of Chapter 6 like MRFS, EVI, and IGS.

The following strategy-exploration policy is the natural extension of this idea to formations—refuting the best ϵ-formation. At each step, the algorithm determines a minimum-ϵ formation that is a subset of current set of restricted strategies, X. The minimum-ϵ-maximum-τ (MEMT) strategy-exploration policy chooses a strategy that maximizes the gain (τ) to deviating from a minimum-ϵ formation. Thus, MEMT selects a strategy that refutes the current minimum-regret formation, choosing the strategy that maximizes τ as a heuristic. If τ is non-positive, then X contains a formation and, thus, the formation-regret is zero. We can view MEMT as a heuristic policy designed to quickly identify formations, by minimizing formation regret. In doing so, the algorithm may also minimize the regret of a minimum-regret profile. The complete procedure is given in Algorithm 9. The algorithm works in two stages: the first stage selects a minimum-ϵ formation (Chapter 4) and the second stage selects the τ-maximizing strategy for the minimum-ϵ formation.

The minimum-ϵ formation, denoted by X_{min}, may have a τ-maximizing strategy, denoted by s_i, that is already in X. In this case, we place X_{min} in a tabu list, denoted by

T. Subsequent calls to FIND-FORMATION will return the minimum-ϵ formation that is not in T or null, if all formations are in T. This process continues until a τ-maximizing strategy is found that is not in X.

Algorithm 9 MEMT($\Gamma^{*}_{S\downarrow X}$)

```
$T \leftarrow \emptyset$
$X_{\min} \leftarrow$ FIND-FORMATION($\Gamma^{*}_{S\downarrow X}, |X|, T$)
$\widehat{s_i} \leftarrow$ NULL
while $\widehat{s_i}$ is NULL and $X_{min}$ is not NULL do
    $s_i \leftarrow$ FIND-TAU($\Gamma^{*}_{S\downarrow X}, X_{\min}$)
    if $s_i \notin X$ then
        $\widehat{s_i} \leftarrow s_i$
    else
        $T \leftarrow T \cup \{X_{\min}\}$
        $X_{\min} \leftarrow$ FIND-FORMATION($\Gamma^{*}_{S\downarrow X}, |X|, T$)
if $\widehat{s_i}$ is NULL then
    $\widehat{s_i} \leftarrow$ FIND-TAU($\Gamma^{*}_{S\downarrow X}, X$)
return $\widehat{s_i}$
```

Consider the symmetric game given in Table 7.1. *How would MEMT and BR select strategies for exploration in this game?* Assume that, at the current step, we have explored strategies a and b. Thus, $X = \{a, b\}$. The BR policy selects a best response to the minimum-regret constrained profile in X. If γ is small ($\gamma < 1/4$), then c is the best response to the minimum-regret constrained profile, where each player plays b with probability $\frac{1-2\gamma}{2}$. Therefore c is selected by BR. Notice that there is a single Nash equilibrium in the base game, where players play a with probability 0.5 and d with probability 0.5. Therefore, BR requires an additional step before a base-game Nash equilibrium is found.

The MEMT policy chooses a strategy that maximizes the regret of a minimum-regret ϵ-formation—the ϵ-formation is a subset of X. The regret (see Definition 23) of $\{a, b\}$ is $1+\gamma$, the regret of $\{a\}$ is $1-\gamma/2$, and the regret of $\{b\}$ is $1+\gamma$. Thus, the minimum-regret ϵ-formation is $\{a\}$, and the τ-maximizing strategy is d. Therefore, d is selected by MEMT, forming a restricted game with strategy set $\{a, b, d\}$. Notice that $\{a, d\}$ is a formation, therefore any Nash equilibrium found in the restricted game with strategy set $\{a, d\}$ is a

Nash equilibrium in the base game.

	a	b	c	d
a	$(0,0)$	$(\gamma, 1-\gamma)$	$(0, 1-\gamma)$	$(1-\gamma/2, 1-\gamma/2)$
b	$(1-\gamma, \gamma)$	$(0,0)$	$(0, 1+\gamma)$	$(0, 1-\gamma/2)$
c	$(1-\gamma, 0)$	$(1+\gamma, 0)$	$(0,0)$	$(0, 1-\gamma/2)$
d	$(1-\gamma/2, 1-\gamma/2)$	$(1-\gamma/2, 0)$	$(1-\gamma/2, 0)$	$(0,0)$

Table 7.1: Example two-player, four-action symmetric game.

7.4 Experiments

I evaluate MEMT on two market games employed by Schvartzman and Wellman for testing their strategy exploration policies. The first is a four-player empirical game simulating a continuous double auction (CDA) scenario (Schvartzman and Wellman, 2009b). The CDA game is symmetric with 13 strategies. The second is based on the Trading Agent Competition Travel (TAC/Travel) game (Wellman et al., 2007). Schvartzman and Wellman use a symmetric two-player hierarchical reduction of a 35-strategy empirical TAC/Travel game for testing.

I consider two evaluation metrics: *expected minimum profile regret* and *expected minimum formation regret*. The expected minimum profile regret of an augmented restricted game is simply the regret of a selected profile—the same as that proposed by Schvartzman and Wellman (2009a). Schvartzman and Wellman select a Nash equilibrium of the restricted game, whereas I select an MRCP. By construction, MRCPs reveal the minimum profile regret of a restricted game, however in the experiments the difference in regret between the MRCPs and NEs is negligible. I proceed with the minimum profile-regret analysis and later return to the minimum formation-regret analysis.

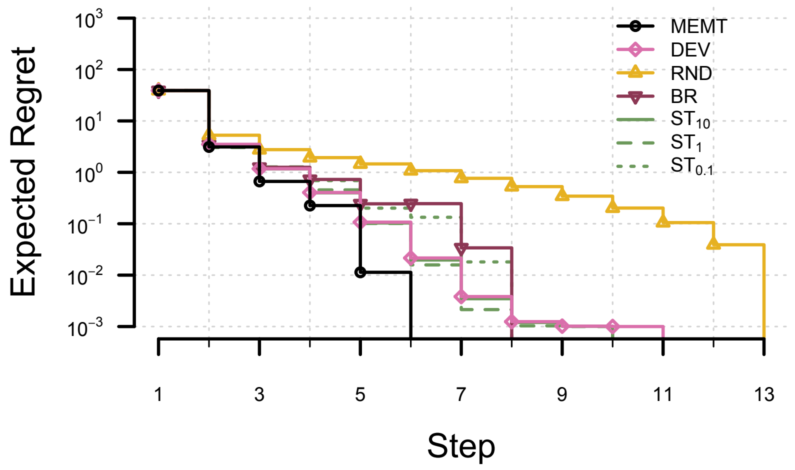


Figure 7.2: The expected minimum profile-regret in the empirical CDA game.

Minimum profit-regret analysis

Using the same experimental setup as Schvartzman and Wellman (2009a), I analyze the MEMT policy by tracing the sequence of augmented restricted games generated by MEMT, starting from each of the 13 strategies of the CDA game. At each step k, I measure the expected regret of an MRCP of $\Gamma^{*}_{S\downarrow X^{(k)}}$. I use a regret tolerance of 10^{-3} when analyzing the traces. If a profile with base-game regret below this level is found, the trace terminates. Figure 7.2 shows the expected minimum profile-regret of MEMT and those policies given by Schvartzman and Wellman, averaged over the 13 traces, on a logarithmic scale for regret. MEMT requires six steps to reach the tolerance threshold in the worst case, but reaches the threshold in 3.84 steps on average. This differs substantially from the improving-deviation policies, where it takes eight steps to reach the 10^{-3} level in the worst case.

Continuing with the same experimental process, I analyze the policies starting from each of the 35 strategies of the TAC/Travel game. Figure 7.3 shows the expected minimum profile-regret averaged over the 35 traces, on a logarithmic scale for regret. In contrast to the CDA experiment, here MEMT and BR display similar quality. The BR and MEMT policies require seven and eight steps, respectively, to reach the tolerance threshold in the worst case, but reach the threshold in 4.74 and 4.86 steps on average.

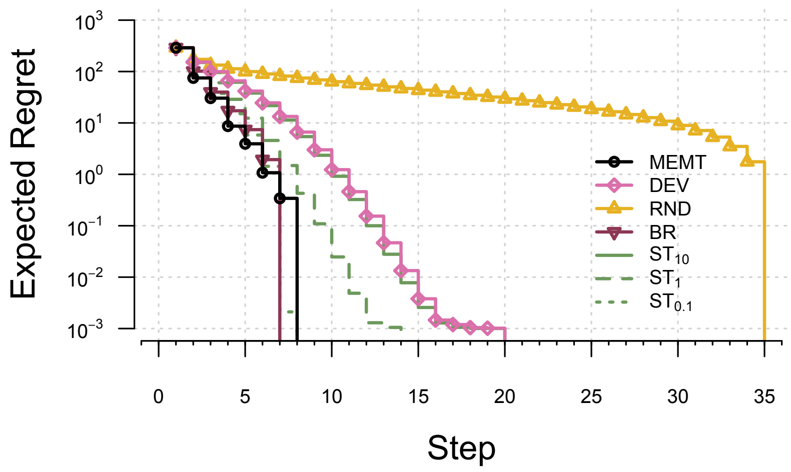


Figure 7.3: The expected minimum profile-regret in the empirical TAC/Travel game.

Minimum formation-regret analysis

I evaluate MEMT and BR, the only other deterministic policy, on the formation-regret measure for the CDA and TAC/Travel games. As in Section 7.4, I use a regret tolerance of 10^{-3} when analyzing the traces. The BR policy is representative of the improving-deviation policies, whereas MEMT is a policy specifically designed to minimize expected formation regret. Like the profile-regret analysis of the CDA game, I find that MEMT and BR required six and eight steps, respectively, to reach the tolerance threshold in the worst case, but reach the threshold in 3.84 and 4.30 steps on average. One explanation for this quick convergence is the existence of a 2-strategy primitive formation. This is the smallest formation that exists in the game and, therefore, the lower bound on the optimal number of steps to termination. If either support of the smallest primitive formation is the starting strategy, both policies terminate in two steps. Of the 13 starting strategies, ten terminate in four steps or fewer for MEMT and nine for BR, respectively. In cases where the best-response strategy supports the minimal profile-regret and the minimal formation-regret, both policies behave similarly. This occurred frequently in the CDA game, potentially due to its small primitive-formation.

Unlike the CDA game, the TAC/Travel game does not have a small primitive formation.

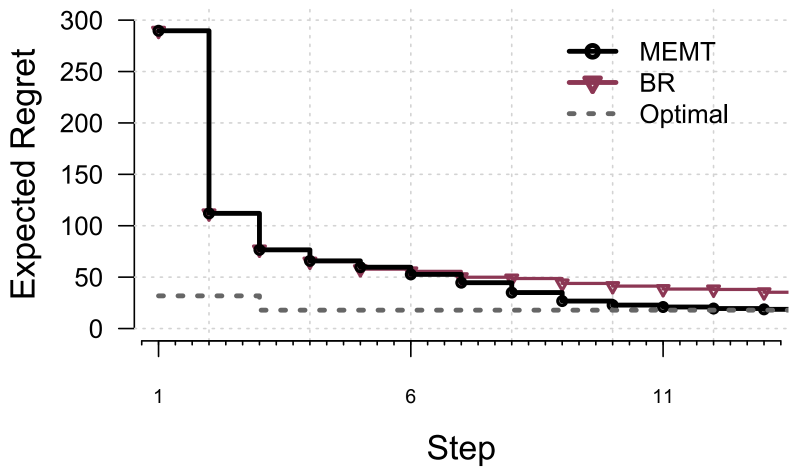


Figure 7.4: The expected minimum formation-regret in the empirical TAC/Travel game.

In fact, the smallest primitive formation has 27 strategies. Figure 7.4 shows the expected minimum formation regret averaged over the 35 traces up to a maximum of 16 steps. The dotted line shows the formation regret of the optimal policy—the sequence that minimizes formation regret. The regret of MEMT nearly converges in mean to the regret of the optimal policy after 15 steps, whereas BR is slower to converge with an expected regret of approximately 1.8 times that of MEMT after the same number of steps.

7.5 Discussion

I present an algorithm for a modified version of the strategy exploration problem defined by Schvartzman and Wellman (2009a). In the modified problem, policies generate a sequence of augmented restricted games. The augmented restricted games allow for exact calculation of profile and formation regret with respect to a base game. In the original strategy exploration problem, Schvartzman and Wellman identified pathological cases where the minimum profile regret increases as additional strategies are explored. With augmented restricted games, this pathology cannot occur because the profile that minimizes base-game regret is selected by the policies, rather than the one that minimizes regret with respect to the restricted game.

I find evidence that formation-finding policies like MEMT perform well on the profile regret measure when small primitive formations exist and are comparable to the best improving-deviation policies when they do not. I also find MEMT is superior according to the formation-regret measure on both game instances. In the large game analysis, the mean expected regret of MEMT converges to the optimal policy much faster than BR—a representative from the improving-deviation family of policies.

Finally, while MEMT is a strategy exploration policy, it can easily be modified for more general profile selection problems—problems with fewer constraints on evaluation. For instance, at each step we can define a candidate restricted game and select a strategy to refute that game as a formation. This policy uses a MEMT-like selection criterion, however MEMT—being a strategy exploration policy—evaluates the restricted game consisting of all previously selected strategies. This is probably inefficient; allowing the policy to evaluate arbitrary restricted games saves us from evaluating unnecessary profiles. This is similar to the policy implicitly employed in Chapter 5, which avoided evaluating a large portion of the profile space.

Chapter 8
Generalization Risk in Empirical Games

Experimental analysis of agent strategies in multiagent systems presents a tradeoff between granularity and statistical confidence. Collecting a large amount of data about each strategy profile improves confidence, but restricts the range of strategies and profiles that can be explored. In Chapter 2, I review Monte Carlo methods for efficiently estimating the parameters of an empirical game model, where we specify the form of the model—for instance, the set of parameters—being estimated. In this chapter, I investigate how we arrive at that form.

I propose a flexible approach, where we construct multiple game-theoretic models for the same underlying scenario (observation dataset). From these candidates, we select a model to represent the scenario. The prospect of incorrectly selecting an empirical model is termed *generalization risk*, and the generalization risk framework I describe provides a general criterion for empirical modeling choices—such as adoption of factored strategies or other structured representations of a game model. I propose a principled method of managing generalization risk to derive the optimal game-theoretic model for the observed data in a restricted class of models. Application to a large dataset generated from a trading agent scenario confirms the method's efficacy.

8.1 Sources of Risk in Empirical Games Models

I consider two sources of risk when modeling empirical games. The first source arises from our choice of the heuristic strategy set, which can be constrained in size by limitations in our computational resources. If the heuristic strategy set is a proper subset of the strategy space, then there is a chance that a useful strategy is omitted. For instance, in TAC/SCM we are limited to analyzing a handful of strategies even though the space of possible strategies is infinite. This risk is endemic in modeling MAS scenarios and can be mitigated only partially by expanding our computational budget.

A second source of risk arises from the analysis of the available observations. Once the simulator and strategy sets are defined, then observations can be collected and an empirical game model estimated from the observation data. The empirical game model provides an estimate for the utility of playing a profile within its profile space. In the previous chapters, the strategy space of the empirical game is assumed identical to that of underlying simulation. I relax this constraint, allowing models where the strategy sets or even players do not correspond precisely to the base notions defined by the simulator. For instance, an empirical game model may treat two strategies that are distinct for the simulator as interchangeable in its own strategy space. This coarsens the model the empirical game uses to predict payoffs, reducing the model's complexity compared to the finer-grained strategy space of the simulator.

Entertaining multiple candidate models of varying complexity provides useful flexibility. A more complex model may capture observations better than a simpler one, however it may also be more susceptible to fitting spurious information in the observations. I term this the *generalization risk* associated with an empirical game model. In the game-theoretic context, the consequence of incorrect generalization could be that profiles that appear stable are actually unstable, or vice versa. Thus, I develop a framework that allows us to compare candidate models and make an appropriate selection based on the model's predictive power. To illustrate the flexibility of this approach, I investigate *equivalent strategy models*, a class of empirical game models that takes an existing strategy space and transforms it by introducing equivalence classes amongst strategies.

8.2 Quantifying Generalization Risk

In this section, I quantify the generalization risk of estimating (*fitting*) the parameters of the embedded utility function, u^{Γ}, to an observation set Θ generated by simulation. Because we may be able to fit multiple models, we require some criterion for selecting among them. In particular, we would like to be able to evaluate and compare the *goodness of fit* for different models so that we may identify which is most useful for analysis.

A standard measure of loss for a statistical model is the mean of squared errors with respect to the data. In this context, I define the loss function $\mathscr{L}$ of a candidate empirical game model by

$$\mathscr{L}(\mathcal{E}|\Theta) = \frac{1}{|\Theta|} \sum_{\theta \in \Theta} \left[\tilde{u}^{\mathcal{E}}(\theta.s) - \theta.\pi\right]^T \left[\tilde{u}^{\mathcal{E}}(\theta.s) - \theta.\pi\right],$$

where $\theta.s$ and $\theta.\pi$ are the joint strategy and payoffs comprising the observation θ. We endeavor to find an empirical game model that minimizes the expected loss $\mathbb{E}[\mathscr{L}(\mathcal{E}|\Theta)]$, where Θ is the random observation set generated from a simulator. Note that because we do not know the true distribution of Θ, we must estimate the expected loss using an existing observation set. In the experiments, I use cross-validation on the observation set to construct this estimate. I outline the k-fold cross-validation procedure in Section 8.3.

8.3 Cross-Validating Empirical Game Models

In order to compare the generalization risk of differing empirical game models, we construct an estimate for the expected loss $\mathbb{E}\mathscr{L}$. I calculate this estimate using a cross-validation technique known as k-fold cross-validation (Stone, 1974).

Define $\mathcal{E}(h, \Theta)$ to be the empirical game model within model class h that minimizes loss with respect to a set of observations: $\mathcal{E}(h, \Theta) = \arg\min_{\mathcal{E} \in h} \mathscr{L}(\mathcal{E}|\Theta)$. I separate the observation set Θ into k distinct partitions as follows. Let $\Theta^{\hat{s}} = \{\theta \in \Theta \,|s(\theta) = \hat{s}\}$, that is, all of the observations of profile $\hat{s}$. For each $s \in S$, I randomly partition Θ^s into k equally sized groups $\Theta_1^s, \ldots, \Theta_k^s$. For a given set $\Theta_i = \cup_{s \in S} \Theta_i^s$, we define $\Theta_{-i} = \Theta \setminus \Theta_i$. Let $\hat{\mathscr{L}}(h|\Theta)$ denote the estimated expected loss given some model class h, defined as follows

$$\hat{\mathscr{L}}(h|\Theta) = \frac{1}{k} \sum_{i=1}^{k} \mathscr{L}(\mathcal{E}(h, \Theta_{-i})|\Theta_i).$$

I refer to $\hat{\mathscr{L}}(h)$ without the Θ parameter when the context is clear.

8.4 Partially Ordered Game Models

In Section 8.2, I use the empirical utility model, $\widetilde{u}^{\mathcal{E}}$, to evaluate the loss of a candidate empirical game model. Here, I introduce a partial order over game model classes. Let h be a set of empirical game models for a given $\mathfrak{G}$. I call h a model class for $\mathfrak{G}$. For instance, we can define a model class by fixing $N^{\mathfrak{G}}$, $S^{\mathfrak{G}}$, N^{Γ}, S^{Γ}, ϕ, and μ while letting u^{Γ} vary. Let $\mathcal{H}$ be a set of models classes under consideration for a simulator.

Definition 27 (Model expressiveness)**.** *For $h, \hat{h} \in \mathcal{H}$, h* expresses *$\hat{h}$ if for every $\hat{\mathcal{E}} \in \hat{h}$ there exists an $\mathcal{E} \in h$ such that $\forall s \in S \quad \widetilde{u}^{\mathcal{E}}(s) = \widetilde{u}^{\hat{\mathcal{E}}}(s)$.*

I use $\hat{h} \preceq h$ to denote that h *expresses* $\hat{h}$ and $\hat{h} \prec h$ to denote that h is *more expressive* than $\hat{h}$, that is, h expresses $\hat{h}$ but $\hat{h}$ does not express h. The pair $(\mathcal{H}, \preceq)$ is a partially ordered set.

Consider two model classes, h and $\hat{h}$, where $\hat{h} \preceq h$. We can decompose the expected loss of each class into bias squared and variance components (for an example, see Geman et al., 1992). Because h is more expressive, we expect the bias of h to be smaller than that of $\hat{h}$. However, we expect the variance of h to be larger than that of $\hat{h}$. Therefore, the tradeoff between the bias and variance components of each class determines the overall expected loss. The heuristic search algorithms defined in Section 8.6 choose model classes to evaluate based on this implicit tradeoff.

8.5 Equivalent Strategy Models

In order to motivate the introduction of *equivalent strategy models*, we return to the example game in Table 1.1 of Chapter 1. In mean utility, strategies (a, A) and (a, B) are equivalent, as well as, strategies (b, A) and (b, B). Therefore, we can reduce the size of our empirical model by considering two strategies instead of four—thus 4 profiles instead of 16. In a factorial design methodology, all possible profiles can be evaluated, and if conducted naively, a great deal of effort will be spent analyzing equivalent strategies.

In the example, the pairs of strategies have exactly the same mean utilities. However, strategies in complex scenarios—such as in TAC/SCM—may be defined by a very large parameter space in which the effects (on utility) of modifying the parameters are slight or imperceptible. Additionally, modelers may not know *a priori* if there are equivalent strategies. Not identifying these equivalent strategies increases the generalization risk—the variance term increases with no corresponding decrease in the bias term—when compared to a model class that equates those strategies. I introduce a game model that uses the concept of equivalent strategies to form a reduced game.

Definition 28 (Equivalent Strategy Model)**.** *An equivalent strategy empirical game model $ESG = \langle \mathfrak{S}, (\sim_i), (u_i) \rangle$ constitutes a model for an empirical game where, for each player i, $\sim_i$ is an equivalence relation on $S_i^{\mathfrak{S}}$ that forms equivalence classes $\{[s_i^{\mathfrak{S}}] | s_i^{\mathfrak{S}} \in S_i^{\mathfrak{S}}]\}$ and $u_i : \times_{j \in N}(S_j / \sim_j) \rightarrow \mathbb{R}$ for each player i such that*

- $\phi(s^{\mathfrak{S}}) = ([s_j^{\mathfrak{S}}])_{j \in N^{\mathfrak{S}}}$
- $\mu(u(\ \phi(s)\ )\ ) = u(\ \phi(s)\ ).$

The embedded game is given by $\langle N^{\mathfrak{S}}, (S_i / \sim_i), (u_i) \rangle$, where $(S_i / \sim_i)$ is the set of equivalence classes on the strategy set S_i—also called the quotient set.

The *equivalent strategy model* forms equivalence class $[s_i]$ from each relation $\sim_i$ where player i may select any element of the equivalence class with the same result. Thus all elements in $[s_i]$ are equivalent. In other words, whenever a strategy s_i is observed in a simulation profile s, we replace it with the representative strategy $[s_i]$. Given some observation set Θ, we can estimate $u_i(\phi(s^{\mathfrak{S}}))$ by the sample mean of the observed payoff set $\{\pi_i|(\pi, s) \in \Theta \text{ and } \phi(s) = \phi(s^{\mathfrak{S}})\}$. Let $\sim$ be equal to $(\sim_i)_{i\in N}$. We can impose the following expressiveness partial order:

$$\hat{\sim} \preceq \sim \Leftrightarrow \forall i \in N\ \forall [s_i] \in (S_i/\sim_i)\ \exists [\hat{s}_i] \in (S_i/\hat{\sim}_i)\ \ [s_i] \subseteq [\hat{s}_i].$$

In other words, every equivalence class under $\sim$ is a subset of an equivalence class under $\hat{\sim}$. By dropping the player parameterization of the equivalence relation in Definition 28, we can define a similar notion of an *equivalent strategy symmetric game*.

	C
A	$\alpha, 0$
B	$0, 0$
$\hat{B}$	$0, 0$

Table 8.1: Simple duplicate game.

Consider another example, the two-player game specified in Table 8.1. The column player has a single action, C, and the row player has three actions: A, B, and $\hat{B}$. Suppose that a simulator modeling this scenario adds zero-mean, unit-variance Gaussian noise to the row player's score. Discounting noise, the strategies B and $\hat{B}$ are equivalent. When α is small A, B, and $\hat{B}$ are approximately the same. We have five basic ESM partitions: *A, B, $\hat{B}$*; *A, $\{B, \hat{B}\}$*; *$\{A, B\}$, $\hat{B}$*; *$\{A, \hat{B}\}$, B*; and $\{A, B, \hat{B}\}$. Clearly, we would like to discover that B and $\hat{B}$ are equivalent. Additionally, for some settings of α, we would like A, B, and $\hat{B}$ to be considered equivalent, if doing so minimizes loss. Consider ESMs fit from a single observation of each profile in the Table 8.1. Because we know the noise distribution, we can compute $\mathbb{E}\mathscr{L}$ analytically. Each payoff observation is a χ-distributed random variable. The expected loss for each partitioning is given in Table 8.2.

Given the expected loss, the optimal model for the game should consider A, B, and $\hat{B}$ equivalent strategies when α is small relative to the noise variance, and only B and $\hat{B}$

Partition	$\alpha = 1$	$\alpha = 2$
$A, B, \hat{B}$	3	3
$A, \{B, \hat{B}\}$	2	2
$\{A, B\}, \hat{B}$	$5/2$	4
$\{A, \hat{B}\}, B$	$5/2$	4
$\{A, B, \hat{B}\}$	$4/3$	$11/3$

Table 8.2: The expected loss, $\mathbb{E}\mathscr{L}$, for various partitions and settings of α for the game in Table 8.1.

equivalent otherwise. Note that in all cases, B and $\hat{B}$ are equivalent in the selected empirical game model, as desired, and when α varies we have a strict decision criterion which allows us to decide what magnitude of payoff differences should distinguish strategies. Of course, in practice we do not have access to the true model—only observations—and must estimate the expected loss.

8.6 Model Selection

In this section, I define a procedure for selecting a model class, h, from a hypothesis space, $\mathcal{H}$. This algorithm is similar to the FIND-FORMATION algorithm of Chapter 4. SELECT-MODEL (Algorithm 10) uses two basic subroutines and a priority queue with a maximum size of M to determine the model class with least expected loss. A setting of $M = 1$ gives a best-first search. The two enqueue routines, ENQUEUE-FIRST and ENQUEUE-NEXT, each present novel model classes to the priority queue. I discuss two generic implementations of the SELECT-MODEL algorithm: INCREASING-SELECT-MODEL and DECREASING-SELECT-MODEL.

In the INCREASING-SELECT-MODEL variant, we start with the least expressive model classes and introduce model classes in order of increasing expressivity. The ENQUEUE-FIRST routine (randomly) enqueues one of the minimal elements of $\mathcal{H}$. The ENQUEUE-NEXT routine enqueues the set $\{h' \in \mathcal{H} | h' \succ h$ and $\nexists h'' \in \mathcal{H}\, h'' \succ h' \succ h\}$, which is the set of least expressive model classes that are more expressive than h.

Algorithm 10 SELECT-MODEL($\mathcal{H}$, Θ, ENQUEUE-FIRST, ENQUEUE-NEXT, M)

```
$best \leftarrow null$
$queue \leftarrow$ Empty priority queue with maximum size $M$ ordered by $\hat{\mathscr{L}}$
ENQUEUE-FIRST$(queue, \mathcal{H})$
while $queue$ is not empty and termination conditions are not met do
    $h \leftarrow$ top$(queue)$
    if $best$ is null or $\hat{\mathscr{L}}(best) > \hat{\mathscr{L}}(h)$ then
        $best \leftarrow h$
    ENQUEUE-NEXT$(queue,\ \mathcal{H},\ h)$
return $best$
```

Similarly, in the DECREASING-SELECT-MODEL variant, we start with the most expressive model classes and introduce model classes in order of decreasing expressivity. The ENQUEUE-FIRST routine (randomly) enqueues one of the maximal elements of $\mathcal{H}$. The ENQUEUE-NEXT routine enqueues the set $\{h' \in \mathcal{H} | h' \prec h \text{ and } \nexists h'' \in \mathcal{H}\, h'' \prec h' \prec h\}$, which is the set of most expressive model classes that are less expressive than h.

I conclude this section by discussing the DECREASING-ESM-SELECTION algorithm to select an ESM given a set of observations. The algorithm is a variant of the DECREASING-SELECT-MODEL algorithm.[1] The scenario under consideration is symmetric, therefore the player indexing on the strategy equivalence relation $\sim$ is dropped. I let $\mathcal{H}$ be the set of model classes induced from all possible equivalence relations on the simulation strategy space. Therefore, $(\mathcal{H}, \preceq)$ is a lattice. The ENQUEUE-FIRST routine enqueues the maximum element of the lattice, which is simply the model class where each equivalence class contains only a single strategy. There is a bijection between model classes and equivalence relations for ESMs. For a model class h, let $\sim$ be the equivalence relation that corresponds to h. The ENQUEUE-NEXT routine enqueues a set of model classes, such that each model class is formed by merging a single pair of equivalence classes in $\sim$. Finally, the algorithm terminates if the expected loss of the best class seen thus far is less than the expected loss of any class in the queue.

[1]We chose this variant under the assumption that most strategies in the scenario should not be considered equivalent. It turns out that this assumption is incorrect, however the algorithm works well regardless.

8.7 Equivalent Strategy Model Experiments

In the following experiments, I consider game models over observations taken from the TAC Supply Chain Management scenario. I experimentally evaluate the DECREASING-ESM-SELECTION algorithm on two TAC/SCM data sets. In Chapter 5, I describe the construction of the 2008 candidate data set, as well as the methods used to evaluate each strategy. After initial analysis, five variants of DeepMaize 2008 remained as candidates for the final screening of strategies. These five strategies combined with top strategies from the 2007 tournament analysis comprise the strategy set used to select the DeepMaize 2008 tournament strategy. Table 8.3 lists the strategies used in this analysis.

Label	Description
PH	PhantAgent 2007
TT	TacTex 2007
DM6	DeepMaize 2008 variant 6
DM20	DeepMaize 2008 variant 20
DM24	DeepMaize 2008 variant 24
DM25	DeepMaize 2008 variant 25
DM28	DeepMaize 2008 variant 28

Table 8.3: Strategies used in the TAC/SCM data set.

Each simulation of a specific profile requires 7 processor hours on a cluster of computers (1 hour running simultaneously on 7 different processors). A full six-player, seven-strategy symmetric game has 924 distinct strategy profiles. We use 30 samples per profile for statistical analysis of the deviations and regret. This requires, at minimum, 194,040 processor hours to complete. On a typical day prior to the tournament, there were 700 processor hours available. Our candidate analysis usually begins in April and lasts through the end of June. The full six-player analysis requires approximately three times the available amount of processing power—assuming a zero failure rate. However, using a three-player hierarchical reduction (Wellman et al., 2005b), we can reduce the number of distinct profiles to 84, a feasible size for analysis.

In Section 5.4, I describe the development path for the DeepMaize 2008 agent, which consists of a set of candidate strategies. I repeat capsule descriptions of each candidate strategy here. The designation DM6 corresponds to DeepMaize 2008 candidate version 6. DM6 is a slight modification of the strategy used by the DeepMaize 2007 agent. DM6 replaced the 2007 procurement strategy by mimicking PH's long-term procurement strategy. DM20 is a departure from DM6 that changed the procurement policy of the agent in the early portion of the game. DM24 and DM25 varied the mid-game procurement levels from that of DM6 and DM20. DM28 uses a modified component-price prediction algorithm, but is otherwise identical to DM20. The DM28 component-price prediction algorithm decreased the prediction error of the DM20 predictor by about 1% RMS error (Pardoe and Stone, 2008). While a 1% improvement is appreciable, it was not known whether the improvement would be expressed behaviorally in the relative score of the agent. Using these candidate strategies and a background set of agents that had strong support in the equilibrium analysis of Chapter 5, our team needed to make a decision about which candidate to play in the 2008 tournament.

The first experiment attempts to empirically validate the ESM selection algorithm for a reduced set of strategies, where one of the strategies is a known duplicate of an existing strategy. Because the two strategies are identical, the ESM selected by the algorithm should place the original and the duplicate in the same equivalence class. A reduced observation set is constructed from the observations of profiles supported by strategies PH, TT, and DM20. To introduce a duplicate strategy, I created a new strategy label: DM20C. For all observations involving strategy DM20, I changed the label to DM20C with probability 0.5. I ran the DECREASING-ESM-SELECTION algorithm on the resultant observations, We expect the algorithm to identify DM20 and DM20C as equivalent. Table 8.4 verifies that the algorithm does indeed equate the two strategies. The table illustrates the algorithm as it progresses through various iterations. For each ESM model, I report the expected loss in millions. The single star highlights the best model in each round and the double star highlights the final model selected.

Round	Best	Strategy Space	RMSE
0		PH, TT, DM20, DM20C	3.64
		{PH, TT}, DM20C, DM20	12.93
		{PH, DM20}, TT, DM20C	11.27
		{PH, DM20C}, TT, DM20	10.68
		PH, DM20C, {TT, DM20}	4.30
		PH, {TT, DM20C}, DM20	4.24
	$\star$	PH, {DM20, DM20C}, TT	3.57
1		PH, {DM20, DM20C}, TT	3.57
		TT, {PH, DM20, DM20C}	14.74
		{DM20, DM20C}, {PH, TT}	12.92
		PH, {DM20, DM20C, TT}	4.73
FINAL	$\star\star$	PH, {DM20, DM20C}, TT	3.57

Table 8.4: Model selection on the TAC/SCM duplicate data set.

In the second experiment, I consider the full strategy set from Table 8.3. Because the ESM selected by the DECREASING-ESM-SELECTION algorithm depends on the randomized observation-set partition used by k-fold validation, different ESMs may be returned on different runs of the algorithm. Over thirty runs, five distinct ESMs were selected by the algorithm. Table 8.5 gives the frequency of these ESMs. The modal ESM contained the strategy equivalence classes: PH, TT, DM28, and {DM6, DM20, DM24, DM25}.

Equivalent Strategy Game	Freq
PH, TT, {DM6, DM20, DM24, DM25}, DM28	17
PH, TT, {DM20, DM28}, {DM6, DM24, DM25}	7
PH, TT, DM6, {DM20, DM24, DM25}, DM28	5
PH, TT, DM28, {DM20, DM24}, {DM6, DM25}	1

Table 8.5: ESM frequency table for TAC/SCM data set.

Figure 8.1 displays the modal ESM with the four distinct regions separated by the gray barrier. Three regions contain exactly one strategy, respectively. The fourth central region contains the four base strategies which the ESM identifies as equivalent: DM6, DM20, DM24, and DM25. The solid black lines interconnecting the four strategies represent the relative strengths of the pairwise equivalences as determined by their frequency in Table 8.5. For instance, strategies DM24 and DM25 appeared as equivalent in 29 of the 30 selected ESMs and the relationship is drawn with a thick line. Comparatively, strategies DM6 and DM20 appeared as equivalent in 17 of the 30 selected ESMs and the relationship is drawn with a slim line. The second most frequent ESM equated strategies DM20 and DM28 in 7 of the 30 selected ESMs. I denote the weaker equivalence tendency by the slim, dotted line.

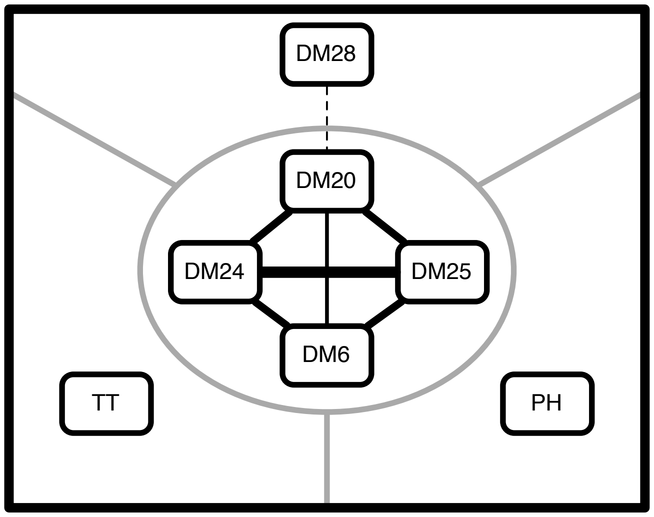


Figure 8.1: Modal ESP for TAC/SCM data set.

For the following experiments, I denote the equivalence class {DM6, DM20, DM24, and DM25} by DM*. I treat the modal ESM as the true ESM of the TAC/SCM simulation. We would like to know how restricting observations to a subset of the strategies affects analysis on the full strategy set. For instance, in designing and analyzing the strategies for a tournament, it often infeasible to sample the entire space of 20 or so candidate strategies, even using the hierarchical reduction technique. Often design and analysis proceed iteratively, adding strategies to a set of small background strategies—similar to the strategy exploration methods of Chapter 7. Bad (irrelevant) strategies are pruned, while those in support of a sample Nash equilibrium or formation (see Chapter 4) are retained. New candidate strategies are added and the analysis process starts a new iteration. There is no guarantee that strategies previously pruned would not be in the support of the new equilibrium.

We would like to know if in a restricted strategy space, the ESM returned by the algorithm is consistent—in terms of the equivalence relation—with the ESM returned by the algorithm given observations from the entire strategy set (the full strategy space). There are two types of errors that may occur: a *false positive* (Type I) and a *false negative* (Type II). A Type I error occurs when strategies that should not be equivalent in the full strategy space are selected as equivalent in the restricted space. Type II errors occur when strategies that should be equivalent in the full strategy space are selected as distinct in the restricted space. For instance, if strategies DM6 and DM28 are in the same equivalence class, the restricted strategy ESM would contain a Type I error. If instead strategies DM24 and DM25 are in different equivalence classes, the restricted strategy ESM would contain a Type II error.

Size		Type I Error Rate			Type II
		PH	DM28	DM*	Error Rate
3	TT	0/5	0/5	0/14	
	PH		0/5	0/14	
	DM28			3/14	
	DM*				1/22
4	TT	0/10	0/10	0/20	
	PH		0/10	0/20	
	DM28			6/20	
	DM*				5/31
5	TT	0/10	0/10	0/15	
	PH		0/10	0/15	
	DM28			3/15	
	DM*				5/21
6	TT	0/5	0/5	0/6	
	PH		0/5	0/6	
	DM28			2/6	
	DM*				3/7

Table 8.6: Type I and II error rates on TAC/SCM data for different strategy set sizes.

I design an experiment to explore the Type I and Type II error rates of the DECREASING-ESM-SELECTION in the TAC/SCM domain. For all of the strategies listed in Table 8.3, we create restricted strategy spaces of sizes 3–6. For each restricted strategy space of size n, I construct observation sets for each of the $\binom{7}{n}$ cases. For each of these observation sets, I run the DECREASING-ESM-SELECTION algorithm. I report the rate of each type of error in Table 8.6, using the modal ESM in Figure 8.1 as the standard for comparison. Note that since the equivalence classes for TT, PH, and DM28 are singletons, there can be no Type II errors for those classes. Because DM* is composed of multiple underlying strategies, we can observe Type II errors when more than one of the constituent strategies is in the restricted strategy set. This occurs with varying rates across the restricted strategy set sizes. For instance, in the restricted size 3 group, of the 22 simulations in which a Type II error could occur, only one contained such a false negative. There were no instances in any of the restricted simulations where TT or PH were incorrectly equated with any of the other strategies. In contrast, DM28 was incorrectly equated with at least one of the DM* strategies 21%, 30%, 20%, and 33% of the time for sizes 3, 4, 5, and 6, respectively. The values seem relatively large, however consider that even when analyzing all seven strategies, 23% of the time DM20 and DM28 were in the same equivalence class.

8.8 Discussion

I propose a framework for comparing empirical game model classes and a family of model algorithms that heuristically select the optimal class from set of candidate classes. For illustration, I introduce a hierarchy of model classes called *equivalent strategy models*. ESMs can model scenarios where duplicate (equivalent or near equivalent) strategies may exist in the simulation space.

I experimentally evaluate DECREASING-ESM-SELECTION, a specialization of the general SELECT-MODEL algorithm for ESMs, on a data set of TAC/SCM observations. Using the algorithm, I select an ESM for TAC/SCM, given the 2008 tournament candidate strategies for our agent. The first test experimentally confirmed that the DECREASING-ESM-SELECTION algorithm identifies identical, relabeled strategies as equivalent. The second test discovered an equivalence relation involving four out of the five DeepMaize candidate strategies. Additionally, the error-rate tests confirm the equivalence classes are consistent across restricted games in the TAC/SCM data.

If ESMs are consistent across restricted games, this property can be exploited by strategy exploration methods. We can analyze small restricted-games for strategy equivalences. If some equivalences are found, they are likely to hold in the base game, thereby reducing the need to test multiple strategies in the same equivalence class. Identifying equivalences can yield a substantial reduction in the required number of observations to fully analyze the scenario. Leading into the 2008 competition, DM20 was identified as a promising strategy. DM28 was a relatively late-breaking addition to the candidate set. Most of the available cycles nearing the start of the tournament were devoted to comparing the difference between DM20 and DM28, which had similar support in a sample Nash equilibrium for the empirical game. DM20 was chosen over DM28 since it had been tested thoroughly, whereas DM28, being a relative last-minute update, had undergone only minimal testing outside of pairwise comparisons. Had the results of the DECREASING-ESM-SELECTION algorithm been available, those cycles could have been devoted to testing other promising candidates.

Chapter 9
Ad Auctions

The emergence of Internet advertising, specifically ad auctions, as a significant commercial success over the past decade (Fain and Pedersen, 2006) has led to increasing interest among academic researchers, manifest in a growing literature and a popular regular workshop on the topic. Both the commercial importance and academic interest were major motivations for introducing a new TAC game in this area. Given that bidding in keyword auctions (employing essentially the same mechanism we incorporate in the game) is a widespread current activity, the prospects for real-world implementation of ideas developed in the research competition are more direct than previous TAC games.

This chapter contains an overview of this new *TAC Ad Auctions* game (TAC/AA), introducing its key features and design rationale. TAC/AA debuted in summer 2009, with the final tournament commencing in conjunction with the TADA-09 workshop. Using tournament and offline simulation data, I perform a strategic analysis of the scenario and find improvements to the standard auction mechanism.

9.1 Simulating Ad Auctions

Despite considerable academic interest in ad auctions, many interesting algorithmic, bidding, and mechanism-design problems remain open (Muthukrishnan, 2008). Designing a *realistic simulator* (Feldman and Muthukrishnan, 2008) is a central component in many of these problems. For instance, Yahoo! researchers (Acharya et al., 2007) developed the

Cassini simulator to evaluate alternative sponsored search mechanisms. The system simulates low-level query and click behavior, publisher ranking and budget enforcement, and other aspects of the sponsored search environment. Cassini allows a rich simulation of user interaction, however the authors report some of its limitations in terms of the advertiser strategy space. For instance, advertisers were not allowed to adaptively change their bids in response to new market conditions. Perhaps most importantly, the Cassini system is not publicly accessible to the research community at large.

Another early predecessor to TAC/AA was the *Pay Per Click Bidding Agent Competition*,[1] designed and organized by Brendan Kitts as part of the ACM EC-06 Sponsored Search Workshop. Participants in this competition managed a live Microsoft AdCenter campaign for a given set of keywords over a 24-hour period. Running the competition with real money and real users over actual sponsored-search interfaces provides a maximal level of realism. In this design, however, I follow the precedent of previous TAC games in developing a simulated environment, where participants interact via a specified interface with a game server running the auctions and generating simulated market events (in this case, search user behavior). This approach provides advantages of repeatability and transparency, which are particularly important for supporting the research goals of this enterprise.

9.2 Sponsored Search

Ad auctions are used by Internet publishers to allocate and price advertising channels. Internet advertising provides a substantial source of revenue for online publishers, amounting to billions of dollars annually. Sponsored search is a popular form of targeted advertising, in which query-specific advertisements are placed alongside organic search-engine results (see Figure 9.1). The placement (position) of ads for a given query, along with the cor-

[1] `http://www.biddingagentcompetition.com`

responding cost (to the advertiser) per click (CPC), are determined through an auction process. Under cost-per-click pricing, both publisher and advertisers bear some of the risk associated with uncertain user behavior. The use of automated auctions addresses the combinatorial problem of quoting an appropriate price (CPC) for each display slot for each distinct query. Advertisers bid for the family of keywords of interest, and competition among them determines the going CPC for each of the available slots on a query-by-query basis.

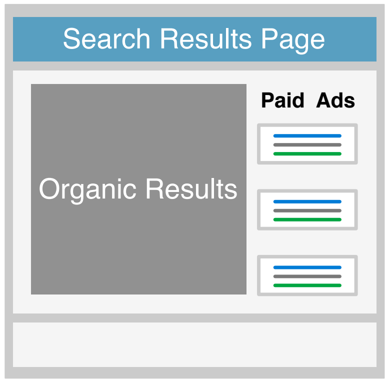


Figure 9.1: Typical search engine results page (SERP).

Given the salience of ad auction mechanisms, a growing number of researchers have started to investigate the *mechanism design* problem faced by search publishers, as well as the strategic problems faced by advertisers. Common to many of the early approaches are stylistic restrictions on the scenario or the full strategic space. Most of the foundational models for sponsored search analysis construct a static game of complete information for a single keyword auction (Aggarwal et al., 2006; Börgers et al., 2007; Edelman et al., 2007; Varian, 2007). This type of analysis has provided a solid conceptual base for researchers to build upon. Significant results include equilibrium characterizations and the discovery that the auctions currently in use by publishers are not truthful. From the static models, extensions have considered dynamic variations that often include some aspect of simulation (Cary et al., 2007; Lahaie and Pennock, 2007; Vorobeychik and Reeves, 2008). TAC/AA continues in this vein by building a richer model space, while retaining many of the useful empirical qualities found in previous TAC scenarios.

9.3 Designing an Ad Auction Game

In TAC/AA, we design a simulation framework that attempts to include many of the interesting strategic aspects of sponsored search auctions, while being repeatable and computationally amenable to empirical analysis. In this framework there are three types of agents: *users*, *advertisers*, and *publishers*. We discuss the behavior of each in turn below, as well as the underlying market that drives the behavior of the user and advertiser agents.

Some important elements of managing an ad campaign are not considered, such as exploration of a large keyword space for high-profitability keywords, or optimizing landing page content to improve the advertiser's quality score. These issues are sacrificed not for lack of interest or value, but rather because we lack useful models to represent them. In the process of developing TAC/AA, we identified three interesting modeling problems, not currently resolved in the sponsored search literature, central to the design of our simulation environment:

- *What drives query generation?*
- *How do advertisers derive value?*
- *Why might keyword auctions be interdependent?*

The process that generates queries is a fundamental component of ad auctions. In addition to defining the query space, we include *query bursts*: large shifts in the number of queries, which advertisers observe in real markets. Uncertainty in the volume of queries is an important consideration for both the publishers and advertisers.

TAC/AA introduces an underlying retail market scenario. This market scenario defines the value of retail purchases to the advertisers. Unlike many of the earlier models, the advertiser value-per-click is not constant. This formulation imposes a keyword-value interdependency based on the query and conversion processes of user behavior. In most other models, interdependency is achieved by exogenous budgets. In reality, the observed budgets—for instance, the daily budgets submitted by advertisers to major search engines—are not hard constraints for the advertisers. For instance, the management of a self-financing firm may impose spend limits on its advertising department to control risk. These constraints are exogenous from the advertising department's perspective, but not from the firm's perspective—in TAC/AA, players are modeled as firms. Thus, spend limits are part of an advertiser's strategy, not an exogenous constraint on it.

A full description of the TAC/AA scenario is provided in the specification document (Jordan et al., 2009). Here, we discuss some of the key modeling choices used in TAC/AA, providing design rationales and comparing to related literature where applicable.

Defining the Market

In the TAC/AA scenario, users search for and potentially purchase components of a home entertainment system. There is a set $\mathcal{M}$ of manufacturers in this market, each of whom produce a set of component types $\mathcal{C}$. The manufacturers and components found in TAC/AA are shown in Table 9.1. The set of products $\mathcal{P}$ is simply $\mathcal{M} \times \mathcal{C}$, therefore there are nine distinct products $p = (m, c)$ that are uniquely identified by manufacturer m and component c. Advertisers represent retailers who deal in these products. Each user has an underlying preference for one of the nine products. The advertisers use the ad auctions to attract user attention to their offerings, in an attempt to generate sales.

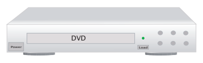
$\mathcal{M}$		$\mathcal{C}$	
	Flat		TV
	Lioneer		Audio
	PG		DVD

Table 9.1: Manufacturers and components in the retail market.

Advertisers each have a distribution process that constrains their ability to deliver products to purchasing users in a timely manner. A user's decision to purchase is influenced by how constrained the advertiser is in making its deliveries. This induces a non-linearity in the value of a click to the advertiser.

Publisher Behavior

Publishers provide the mechanism through which advertisers interact in sponsored search auctions. This includes defining the slots over which advertisers bid, the mechanism that ranks and prices the displayed ads given the bids, and reserve prices that constrain the bids of displayed ads. The value of the slots to the advertisers and the publisher is largely determined by user behavior. This in turn requires advertisers and publishers to construct a model of user behavior in order to optimize their respective objectives. Each of these components of publisher behavior and the associated existing research are discussed subsequently.

Slot positions

When a user queries a publisher, the publisher returns a set of ads. In typical sponsored search auctions, the ads are returned in some significant order (see Figure 9.2). The position of the ad connotes some relative value. This relative value is inferred from the disparity in click-through rates across positions. For example, ads positioned towards the top of the results page usually have higher click-through rates, all else considered. Some search engines divide slots into two regions. One region is considered promoted or premium and is somehow distinguished from the other ad slots. In TAC/AA, we incorporate a similar notion with two types of slots: *regular* and *promoted*. Ads in promoted slots receive an odds bonus in click-though rate.

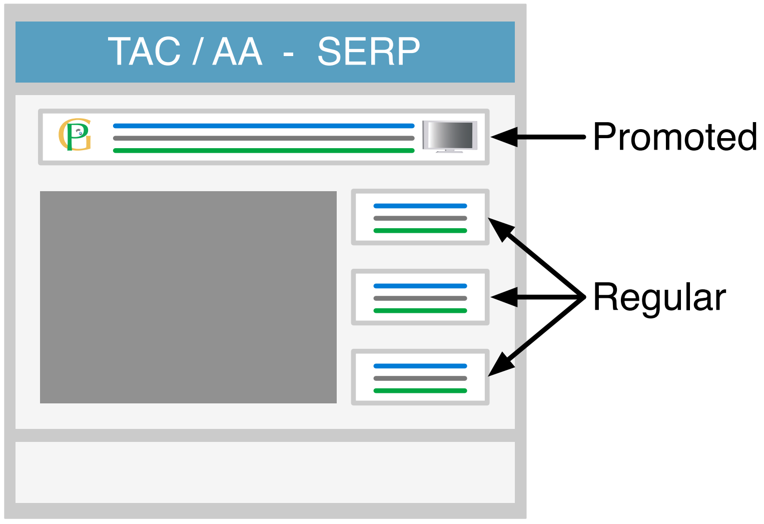


Figure 9.2: TAC/AA search engine results page (SERP).

Ranking ads

In general, each ad auction matches ads with available slots. This type of generality is appealing and general matching mechanisms have been applied to sponsored search by Aggarwal et al. (2009). However, given the natural ordering of slots, mechanisms that fundamentally incorporate this order are often used in practice as well as research.

Two ranking mechanisms have been prominent in the analysis of sponsored search:

- **Rank by Bid:** advertisers are ordered according to their bid b_q for a given query.
- **Rank by Revenue:** advertisers are ordered according to the product of their click-through rate and bid, $e_q b_q$, for a given query.

Lahaie and Pennock (2007) introduce a family of ranking algorithms that can interpolate between *rank-by-bid* and *rank-by-revenue*. The family is parameterized by a *squashing parameter* χ. Advertisers are ranked according to $(e_q)^\chi b_q$, which we term an advertiser's *score*. Notice that a setting of $\chi = 0$ is equivalent to rank-by-bid and a setting of $\chi = 1$ is equivalent to rank-by-revenue. The ranking method in TAC/AA uses the Lahaie and Pennock parameterization. The squashing parameter is announced at the beginning of the simulation, so that advertisers can condition their strategy on it.

Pricing clicks

In sponsored search, a slot is assigned a *cost per click* (CPC) that is determined by an auction. When a user clicks on the ad in the slot, the advertiser is charged the CPC amount. Edelman et al. (2007) describe the two basic pricing mechanisms used in sponsored search auctions.

- **Generalized first-price (GFP):** the CPC for a slot is set to the price bid by the winner of that slot.
- **Generalized second-price (GSP):** the CPC for a slot is set to the minimum price the winner of that slot needed to pay to keep the slot.

Let $b^{(i)}$ be the bid of the winner of the i^{th} position and $e^{(i)}$ be the click-through-rate of advertiser i. Using the Lahaie and Pennock parameterization, the bidder pays

$$b^{(i+1)} \left(\frac{e^{(i+1)}}{e^{(i)}} \right)^{\chi}$$

under GSP.

The auctions introduced by Overture in 1997 used GFP. Edelman et al. report one of the effects of GFP to be volatile prices. Under GFP, advertisers inevitably want to change their bid given the current setting of other-agent bids, which produces a price instability actually observed in such auctions. In practice, most publishers now use GSP, and TAC/AA adopts this pricing rule as well. With GSP, advertisers have less cause to frequently adjust prices, because they are already paying the minimum price for the slot given the other advertisers' bids.

Setting reserve prices

The publisher in TAC/AA employs a generalized second-price pricing model with *reserve scores*. To be allocated a slot, an advertiser's score must meet a certain threshold ρ, called a reserve score. The publisher in TAC/AA uses two reserve scores: ρ_{pro} for promoted slots and ρ_{reg} for regular slots. If an advertiser's score in a (candidate) promoted slot falls between ρ_{reg} and ρ_{pro}, the advertiser is allocated a regular slot.

Let $e_{q,(p)}$ be the baseline click probability of the ad in the p^{th} position, and $b_q^{(p)}$ the bid by the advertiser ranked in that position (we take $b_q^{(p)} = e_{q,(p)} = 0$ if there is no such bid). The cost-per-click (CPC) for position p of the auction for query q is determined by the *effective score* of the next position, which we denote by

$$score_{\mathit{eff}}(p) = \begin{cases} \rho_{(p)} & \text{if } e^{\chi}_{q,(p)} b_q^{(p)} \geq \rho_{(p)} \geq e^{\chi}_{q,(p+1)} b_q^{(p+1)} \\ e^{\chi}_{q,(p+1)} b_q^{(p+1)} & \text{otherwise,} \end{cases}$$

where $\rho_{(p)}$ is ρ_{pro} if p is a promoted slot and ρ_{reg} otherwise. The CPC is given by the minimum the p^{th} advertiser needed to bid to beat this effective score,

$$\frac{score_{\mathit{eff}}(p)}{e^{\chi}_{q,(p)}}.$$

Reserve prices in ad auctions are used for *revenue maximization* and *ad quality control*. Abrams and Schwarz (2008) develop a framework based on the *hidden costs* advertisers impose on users.[2] Abrams and Schwarz construct an efficient mechanism by modifying the bids by the hidden costs. Even-Dar et al. (2008) describe a set of VCG payment modifications that incorporate advertiser-specific minimum bids. One of the payment modifications offsets bids by the minimum reserve prices. Using the Abrams and Schwarz mechanism, Even-Dar et al. show that the auction is efficient and truthful. The other efficient and truthful VCG payment adjustment Even-Dar et al. introduce is *virtual values*. These virtual values essentially become *reserve scores*, where an advertiser's score is the product of its bid and click-through rate. Unlike the more general Even-Dar et al. model, the reserve price model of TAC/AA applies a uniform reserve score across advertisers for a given query. However, the reserve score can be converted into an advertiser-specific reserve price

[2]In their model, hidden costs are related to the change in future revenue due to a user clicking on an advertiser's ad. For instance, dissatisfied users are less likely to click on future ads.

by adjusting for the advertiser's individual click-through rate.

Unknown user behavior

The behavior of users is not known *a priori* to publishers or advertisers. For instance, publishers may view the number of queries per day as a stochastic variable. The distribution may be influenced by many latent variables. Dealing with this type of uncertainty is an important part of a publisher's mechanism.

Recent research has explored various online algorithms for allocating ad slots to advertisers given a random sequence of queries. This problem has been considered with advertiser budgets (Mehta et al., 2007; Mahdian et al., 2007; Muthukrishnan et al., 2007; Goel and Mehta, 2008) and without (Mahdian and Saberi, 2006; Abrams and Gosh, 2007).

In addition to online algorithms, publishers may try to design optimal mechanisms that use various parameterizations of user behavior. In real markets, these parameters must be learned. This learning process affects the dynamics of the auctions, which in turn affects revenue and efficiency (Wortman et al., 2007). Learning parameters is an especially important part of the publisher mechanism when the query space is large and data is sparse. In TAC/AA this is not the case; the query space is relatively small and users generate a large number of queries for each query each time period. For this reason and simplicity's sake, we just assume the publisher in TAC/AA knows advertiser-specific click probabilities, thus eliminating the need to learn click-through rates. We further assume that the ranking mechanism is fixed, so that learning more detailed user behavior is not relevant to publisher behavior. It is, of course, quite relevant to advertiser behavior.

User Search Behavior

Aggarwal et al. (2008) suggest a framework for analyzing sponsored search auctions in which the *search user* takes a central role. The authors describe the need for a rich probabilistic model of user behavior, specifically once the ads are presented to the user. This corresponds to the *click* and *conversion behavior* presented in the sections that follow. We go even further and suggest that the entirety of user behavior should form the basis for analysis. This includes the definition of the query space over which that users generate queries and the frequency at which they do.

Query behavior

Search queries trigger the ad auctions that are built around them and, thus, understanding and modeling the user query process is of fundamental importance. Much of the early research in ad auctions looked at an instance of a single ad auction or a sequence of auctions all associated with a single query class. This abstracts away the interrelation among queries and the implications this has for bidding.

For instance, advertisers often use *keywords* that match multiple queries. Advertisers must reason about their values for each type of query that a single keyword matches. Moreover, the relative frequencies of queries changes dynamically over time, thus the value of the keyword changes with the distribution. This implies that query dynamics is an important consideration as well, when designing an ad auction simulation.

TAC/AA uses a state-based user model to generate this dynamic behavior (see Figure 9.3). Users progress through various states in order to satisfy their underlying product preferences. The user's state determines the type of query the user generates (see Guo et al., 2009, for related work). At any given time, the population of users is divided into three broad classes: *non-searching* (NS), *searching*, and *transacted* (T). This is similar in nature to the query classifications of Jansen et al. (2008). Non-searching users are currently inactive, generating no queries. The searching users are further divided into *informational* (IS) and *shopping* searchers. The informational searchers seek to gather information about their desired product but not to purchase. The shoppers navigate available ads and possibly transact. Shopping users are further divided by levels of search sophistication[3] (focus): low focus (level 0), intermediate (level 1), and high focus (level 2). The transacted users have satisfied their preferences and thus do not search.

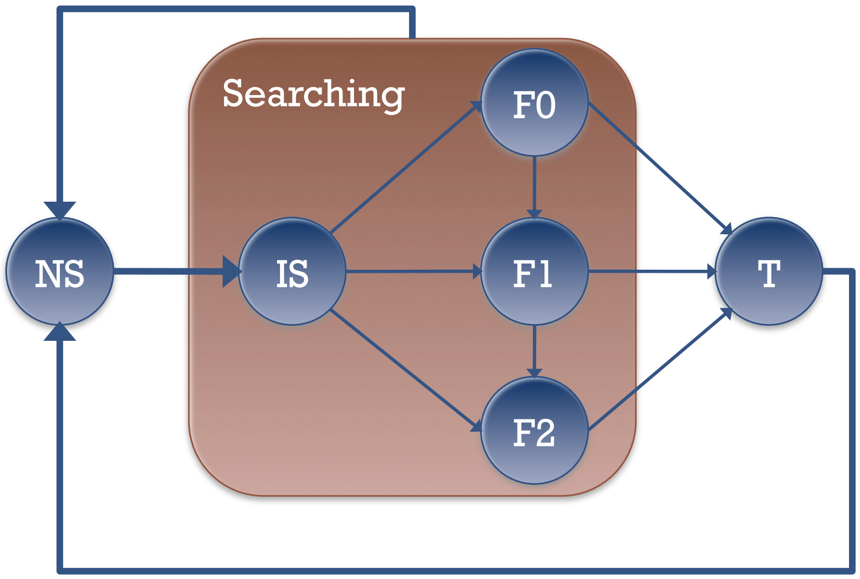


Figure 9.3: User state transition model. Each state also has an implicit self-loop (not shown).

A query consists of a collection of words. In our model, we consider only the six words corresponding to manufacturers and components in the home entertainment market. Each query contains at most two of these words: the user's desired manufacturer and component. For instance, a user with preference *(Lioneer, TV)* may generate a query mentioning *Lioneer*, *TV*, both *Lioneer* and *TV*, or neither. An F0 level query mentions neither a component nor manufacturer. An F1 level query mentions one or the other, but not both. an F2 level query mentions both component and manufacturer. In total, there are 16 distinct queries: 1 F0 query, 6 F1 queries, and 9 F2 queries. A user with a given product preference will generate one of four queries: two possible F1 queries, and one possibility each at F0 and F2.

Each user in a searching state generates a single query per day. An F0, F1, or F2 user

[3]We can also think of these levels as reflecting their degree of knowledge about their own preference.

submits a query pertaining to its level of focus. An informational user selects among the three query types uniformly at random. If an F1 query is selected, the informational user selects between the manufacturer and component with equal probability.

Each user sub-population is modeled as a Markov chain. Most transition probabilities are stationary, with the following exceptions. To model bursts of search behavior, we provide stochastic spikes in the $NS \rightarrow IS$ transition. The transition probabilities from focused search states to state T are also non-stationary, governed by the click and conversion behavior of the user.

Click behavior

Many models have been proposed for user click behavior. The functional forms of the models vary, nevertheless each model gives the probability that an ad at a given position will be clicked. Examples of early models include the Edelman et al. (2007) model an Börgers et al. (2007) model. Edelman et al. (2007) assume each position has an ad-independent click probability—thus, they do not consider the effect of an ad on the probability. In contrast, Börgers et al. (2007) allow for an independent probability for each advertiser-position pair.

Even the Börgers et al. model is not completely general. For instance, it may be that the click probability is *dependent* on the other advertisers and the position of the other advertisers that are allocated slots. Most existing research (implicitly) adopts one of the following models for click probability:

- **Separability:** For each query, the click probability is the product of a *position* and *advertiser effect* (Aggarwal et al., 2006; Edelman et al., 2007; Börgers et al., 2007; Varian, 2007);
- **Cascade (Markovian):** For each query, each ad has a *click probability* given that the ad is viewed, as well as a *continuation probability* that the user will view the subsequent slot (Aggarwal et al., 2008; Kempe and Mahdian, 2008).

The decomposition given by the *separability model* yields a convenient form for the optimization problem the publisher solves, however this model does not appear to be the best predictor of click probabilities. For organic (non-sponsored) links, Craswell et al. (2008) find the *cascade model* to be the best predictor of click probabilities and argue for applicability of their results to sponsored links. The dependency of click probability on the other advertisers is termed an *externality effect*. Gunawardana and Meek (2008) analyzed these effects and found a significant *contextual effect*. In general, Gunawardana and Meek's results suggest that significant externality effects exist and that the assumptions of the separability model do not hold in practice.

As an alternative to the cascade model, Das et al. (2008) propose an extension of the separability model in which the user may convert from at most one of the advertisers. Like the cascade model, this introduces a dependence on the advertisers in the higher slots.

The click model we employ in TAC/AA is a hybrid of the cascade model and the model proposed by Das et al., and also incorporates the underlying product preferences of individual search users. Users in our model proceed as in the cascade model, but stop clicking on subsequent ads when a purchase is made.

In practice, one important focus of *search engine marketing* (SEM) is selecting the ad copy or the text that is displayed in the ad. This process usually involves creating a series of ads and then testing the click-through rates of those ads, known as *split testing*. The TAC/AA click model does not incorporate text directly, however it does include a rudimentary form of ad selection. Ads take one of two forms: *targeted* and *generic*. Targeted ads emphasize a specific product, whereas generic ads do not. Ghose and Yang (2008) discuss the effects of brand and product keywords on click probability. The TAC/AA model incorporates similar effects, but in terms of ad targeting. Compared to the generic ad, users with preference matching the target of a targeted ad click with higher probability, and non-matching users are less likely to click.

Specifically, the click behavior of searching users is modeled by the following parameters:

- an advertiser effect e_q^a for each combination of advertiser a and query class q,
- a targeting effect TE which modifies the probability of clicking targeted ads depend-

ing on whether the user's preferences match the ad target,
- a promotion bonus modifying the click probability for promoted slots, and
- a continuation probability γ_q for query class q.

Given an impression page, also called a search engine results page (SERP), for query q, the user proceeds to sequentially view ads, starting from the first position. For a generic ad viewed from advertiser a, the baseline probability that the user clicks is given by e_q^a. This probability can be modified by two factors. First, the *targeting factor*, f_{target}, applies the targeting effect positively or negatively depending on whether the targeted ad selection matches user preference:

$$f_{\text{target}} = \begin{cases} 1 + TE & \text{if targeted ad, matches} \\ 1 & \text{if generic ad} \\ 1/(1 + TE) & \text{if targeted ad, does not match.} \end{cases}$$

Second, the *promotion factor* f_{pro} applies a *promotion slot bonus* PSB if the ad position is a promoted slot. Promoted slots are placed in a premium location on the page (see **Slot positions**), and therefore enjoy an enhanced click rate. For a regular slot, $f_{\text{pro}} = 1$, and for a promoted slot, $f_{\text{pro}} = 1 + PSB$.

The overall click probability starts with the baseline and gets adjusted based on these factors.

$$\Pr(\text{click}) = \eta(e_q^a, f_{\text{target}} f_{\text{pro}}),$$

where

$$\eta(p, x) = \frac{px}{px + (1 - p)}. \tag{9.1}$$

If the ad is not clicked, or clicked but no purchase is made, the user proceeds to the next ad with continuation probability γ_q.

Conversion behavior

The purchase or conversion behavior of users can arise from various processes (Chen and He, 2006; Athey and Ellison, 2007; Cary et al., 2008; Kominers, 2008). For example, there may be some cost associated with search for the users and the advertisers may have differentiated products and prices. In any case, these models induce some probability that the user will convert.

In TAC/AA, we describe this conversion probability in terms of inventories and backorder delays. This story is meant merely to be suggestive, just one causal explanation for the ultimate effect, which is to impose a diminishing marginal value on clicks. Our conversion model is composed of three factors. One factor is attributed to the state or type of the user. The other two factors are associated with the state of the advertiser and its product specialty, respectively.

Once an ad has been clicked-through, the shopping users convert at different rates according to their focus levels. The probability is a function of several parameters. The baseline conversion probability is given by π_l, for $l \in \{\mathrm{F0}, \mathrm{F1}, \mathrm{F2}\}$. Higher focus level queries convert at higher rates: $\pi_{\mathrm{F2}} > \pi_{\mathrm{F1}} > \pi_{\mathrm{F0}}$.

The second factor captures an effect of constrained distribution capacity. The story is that if the advertisers sell too much product in a short period, their inventories run short and they have to put items on backorder. As a result, shoppers are less inclined to purchase, and conversions suffer. All product sales contribute to the distribution constraint, thus rendering the queries interdependent. Let c_d be the total number of conversions over all products on day d, and W the aggregation window for distribution capacity. The distribution constraint effect is given by

$$I_d = \lambda^{\left(\sum_{i=d-1}^{d-W} c_i - C^{cap} \right)^{+}},$$

where C^{cap} is the critical distribution capacity, beyond which conversion rates decrease. In our scenario, advertisers are assigned one of three discrete capacity levels: $cap \in$

$\{\text{HIGH}, \text{MED}, \text{LOW}\}$.

Finally, we consider the effect of component specialization. For users with preference for a component matching the advertiser's specialization, the odds of converting are increased by a component specialization bonus (CSB), using the formula for odds adjustment (9.1). If the user matches component specialty, $f_{\text{specialization}} = 1 + CSB$, otherwise $f_{\text{specialization}} = 1$. In sum, the overall expression for conversion probability becomes

$$\Pr(\text{conversion}) = \eta(\pi_l I_d, f_{\text{specialization}}).$$

Advertiser Strategy Space

Advertisers in sponsored-search auctions face a complex problem in optimizing their ad campaigns. They contend with dynamic user behavior, uncertainty in publisher policies, and the effects of other competing advertisers. Advertisers control the content of the ads, which ads to display, the bids they place for the ads, and spend limits that bound the cost they can incur. There are also other aspects of campaign management that are important. For instance, optimizing the *landing page*—the page users are directed to when the click on an ad—can dramatically affect conversion rates.[4]

All of these features define the advertiser strategy space, however only a subset of these features are included in TAC/AA. Part of the motivation for excluding some features (in addition to simply limiting scope), such as *landing page optimization*, is that we expect them to be approximately strategically independent and can be studied in a decision-theoretic context apart from other strategic considerations. Features that we believe are strategically *dependent* include setting bids, choosing ads, and setting spend limits. We discuss each of these in turn over the remainder of the section.

[4]Fields such as *information architecture* (IA) and *human-computer interaction* (HCI) that are devoted to improving user experience.

Bidding

Advertisers may specify bids for each of the *query classes*—there are 16 distinct query classes in TAC/AA. This differs with bidding languages that are actually employed by search engines where advertisers bid on *keywords*. Even-Dar et al. (2009) identify the bidding language used by TAC/AA as a *query language* and those used by the search engines as a *keyword language*.[5] In the case of a keyword language, advertisers are forced to implicitly reason about their values over a set of queries. Thus, the selection of keywords becomes a major component of the advertiser's strategy. Various natural language processing and machine learning models have been proposed that attempt to generate or select profitable keywords (Rusmevichientong and Williamson, 2006; Bartz et al., 2006; Abhishek, 2007; Chen et al., 2008). We adopt a query language over the restricted domain of TAC/AA queries for two reasons: query languages are more expressive and the query space in TAC/AA is small.

Choosing ads

In actual sponsored search auctions, advertisers generate the ads that are displayed. The content of the ad relative to the user query can have a dramatic effect on the click-through rate of the ad. Advertisers, or SEM firms managing campaigns on their behalf, typically develop ad content in an iterative manner. First, a set of candidate ads is created and submitted to the publisher for display. Then, some method of testing is used to prune ads that perform poorly. Based on the surviving ads, the advertisers generate additional candidate ads for testing and the process recurs.

The ad content in TAC/AA is specified by the inclusion, or lack thereof, of a specific product. An ad that specifies a product is called *targeted*, whereas an ad that does not is called *generic*. This restricts the set of possible ads and eliminates the content creation

[5]Equivalently, one can view the TAC/AA query language as fixing a coarse partition over a large set of implicit keyword expressions.

aspect of the advertisers' strategies. However, the exploration and exploitation problem of selecting which ad to display for a given query remains.

Setting spend limits

Currently, most publishers allow advertisers to specify an *advertising budget* by which an advertiser can limit the advertising cost or spend for some period of time. Once the advertiser exceeds the limit, the constrained ads will no longer be shown.

Much of the published work on advertiser bidding strategies in dynamic, multi-keyword sponsored search auctions focuses on optimizing return while being constrained by an exogenously specified budget (Kitts and Leblanc, 2004; Zhou and Lukose, 2007; Muthukrishnan et al., 2007; Zhou et al., 2008; Zhou and Naroditskiy, 2008). We believe that in most situations the "budgets" submitted by advertisers to publishers are actually soft constraints on spending. These *daily spend limits* can be used by the advertisers for a variety of purposes—for instance, to protect against a large influx of unprofitable clicks, to guard against the advertisers' uncertainty about the value of those clicks, cash-flow management, or controlling distributed purchasing in a large organization.

Simulating an Advertising Campaign

The TAC/AA game simulates the daily campaigns of eight advertisers over a horizon of 60 simulated days. A high level depiction of the game interaction is show in Figure 9.4. The game flow can be described by considering the game initialization phase and the daily tasks performed by the agents after initialization.

At the beginning of a game instance, the instance-varying user, advertiser, and publisher parameter settings are drawn from their associated distributions. All users are initialized to the non-searching state, and the server simulates virtual days of user activity without advertising, to spread the population across various states. The virtual day initialization is

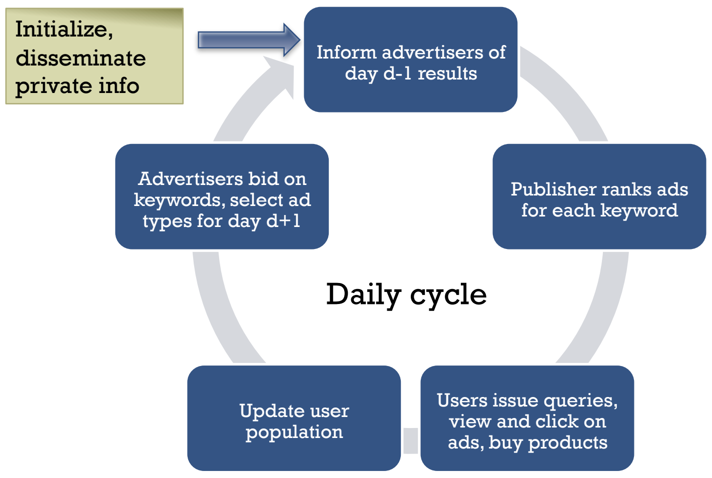

Initialize, disseminate private info
Inform advertisers of day d-1 results
Publisher ranks ads for each keyword
Users issue queries, view and click on ads, buy products
Update user population
Advertisers bid on keywords, select ad types for day d+1
Daily cycle

Figure 9.4: Cycle of activities for day d of a TAC/AA game instance.

an attempt to reduce the impact of *cold start* anomalies. Advertisers learn their product and manufacturer specialization as well as their distribution capacity parameter (they are not told the specialties and capacities of competitors). Finally, the publisher determines the squashing parameter χ and reserve scores, of which the squashing parameter is revealed to the advertisers.

At the beginning of each day d, the daily reports summarizing day $d-1$ activity are delivered to the advertisers. The publisher executes an ad auction for each query class to determine the ad rankings and click prices. Users then issue queries, receive results, consider clicking on ads and purchasing products. The publisher monitors spend limits and reruns ad auctions as necessary. After all searching users have acted, the server updates the population based on the results of the queries, ads, and purchases. Finally, the advertisers submit their bid and ad selection updates to the publisher, for the auctions determining placement on day $d+1$.

At the conclusion of a game, log files are produced that trace the interaction of the agents during the simulation. We provide a log file parser that allows for further post-game analysis of the traces.

9.4 The 2009 Tournament

The TAC/AA specification and server implementation were announced in late 2008, and the first tournament was held in Summer 2009. The 2009 TAC/AA competition had two stages: *qualifying* and *final tournament*. During the qualifying stage agents participated in a round-robin style tournament. Agents passed the qualifying stage by meeting a minimal standard for agent competence. The TAC/AA final tournament was held during the Trading Agent Design and Analysis (TADA) workshop as well as the main IJCAI conference in July 2009. Table 9.2 lists the participating teams' agents and affiliations.

Agent	Affiliation
AstonTAC	Aston University
Bishop	Aston University
CrocodileAgent	University of Zagreb
DNAgents	University of Washington, Tacoma and Ghent University
EPFLAgent	Ecole Polytechnique Federale de Lausanne
McCon	Concordia and McGill Universities
Merlion	Singapore Management University
Mertacor	Aristotle University of Thessaloniki
MetroClick	City University of New York and Microsoft
QuakTAC	University of Pennsylvania
Schlemazl	Brown University
TacTex	U Texas at Austin
UMTac09	University of Macau
UWTAgent	University of Washington, Tacoma
WayneAd	Wayne State University

Table 9.2: Team agents and affiliations for the 2009 TAC/AA tournament.

The final tournament consisted of two rounds: *semifinals* and *finals*. During each round approximately half of the agents were eliminated, while the remaining agents continued on to the next round. In total, fifteen teams participated in the competition with eight moving on to the finals round. The semifinal round consisted of a total of 88 games, while the final round consisted of 80. Teams in the semifinal round played in either 44 or 48 games, while all agents in the finals played in each of the 80 games. Table 9.3 gives the scores for each agent in semifinal and final rounds.

Agent	Finals	Semifinals	Agent	Semifinals
TacTex	79.89	70.69	Mertacor	62.53
AstonTAC	76.28	66.45	UWTAgent	62.32
Schlemazl	75.41	68.93	Merlion	61.27
QuakTAC	74.46	64.14	Bishop	59.31
DNAgents	71.78	68.28	McCon	41.73
EPFLAgent	71.69	68.22	CrocodileAgent	34.55
MetroClick	70.63	63.04	WayneAd	19.19
UMTac09	66.93	65.57		

Table 9.3: TAC/AA-09 tournament participants, with average scores ($K) from semifinal through final rounds.

Each round of the tournament consisted of a set of *blocks* of four games. Within each block, each agent had two games where its C^{cap} value was MED, one game with a value of LOW, and one game with a value of HIGH. The C^{cap} variable is an influential environmental factor that significantly affects an agent's respective score, therefore the blocks enforced a fair distribution of playing time with each value (see the *common random variables* and *antithetic variables* methods of Section 2.1).

9.5 Tournament Analysis

Our understanding of the strategies employed by the participants in the 2009 tournament is limited to observations (simulation traces) from tournament play and the offline analysis of the following section. I perform a *black-box* analysis of the agents in order to gain an understanding of the elementary components of their strategies. I investigate the distribution of advertising cost (spend) over query classes for each of the agents. With respect to a set of basic features, the distribution of advertising spend is the most significant observable behavior distinguishing agents in the 2009 tournament.

Constructing the dataset

As with TAC/SCM, agents may experience network delays or other problems that cause an agent to exhibit non-standard behavior by no fault of their own. A game is scratched in TAC/SCM if, for any agent, there are six or more days (out of 220) in which the server did not receive a message from that agent (as indicated by the game log). The games are shorter in TAC/AA (60 days) and so we adopt a tighter bound. Two or more days without a message triggers a scratch. Of the 168 games in the final rounds, 20 games meet this criterion.

Reducing the variance in profits

I employ the Monte Carlo techniques of Chapter 2 to reduce variance in the scores observed in TAC/AA simulations. In TAC/SCM, we observed a significant correlation between the average level of demand and the profits agents received (see Section 5.3). The derived payoff measure, DAP, adjusted each agent's profit by the same amount. This uniform adjustment occurs because the demand is assumed to affect each agent similarly and the average demand is observed by all agents. Unlike TAC/SCM, there are significant control variables in TAC/AA that are specific to a given agent in a simulation, such as the critical distribution capacity C^{cap}. This introduces an agent-specific dependence in the control variates' score adjustment that is not observed in TAC/SCM. Therefore, the adjustment may change the rank (order of scores) of agents within a simulation.

I use data from the 2009 TAC/AA Final Tournament to calculate the control variates. Of the 168 games in the final tournament, I scratch eight from the semifinal round. Twelve games from the final round matched the scratch criterion. In each case, it was TacTex that caused the scratch, however TacTex's scores were not adversely effected. Thus, I include those games in the test dataset. I consider the set of candidate control variables given in Table 9.4. For each simulation, I calculate the value of the control variables for each of the eight agents. I use the semifinals dataset to fit the regression models and the finals dataset to test. This generates a total of 640 training samples and 640 test samples that I use to calibrate the control variates.

Symbol	Description	Mean
χ	Squashing parameter	0.8
ρ_{reg}	Regular reserve score	0.035
ρ_{pro}	Promoted reserve score	0.2175
C^{cap}	Advertiser critical distribution capacity	400
N^{ps}	Number of advertisers with product specialty	7/9
N^{ms}	Number of advertisers with manufacturer specialty	7/3
N^{cs}	Number of advertisers with component specialty	7/3
e	Mean advertiser effect	0.4
e^{ps}	Product specialty advertiser effect	0.45
e^{ms}	Mean manufacturer specialty advertiser effect	0.425
e^{cs}	Mean component specialty advertiser effect	0.425
γ^{ps}	Product specialty continuation probability	0.55
γ^{ms}	Mean manufacturer specialty continuation probability	0.525
γ^{cs}	Mean component specialty continuation probability	0.525

Table 9.4: TAC/AA-09 candidate control variables with their respective mean values.

Chapter 2 describes the luck-only method of control variates for multiagent simulations. I propose four regression types for the luck-only method, of which I consider two here: *strategy-independent* and *strategy-dependent*. In the strategy-independent method, we regress the payoffs against the control variables *without* a strategy indicator. In the strategy-dependent method, we regress the payoffs against the control variables *with* a strategy indicator as an addition factor in the regression. The intuition behind the strategy-dependent method is that, in general, agents are likely to score differently on average due to some inherent difference in strategic quality. If we do not account for this, the regression will use correlations in the control variables to "explain" the variance in scores. In some cases, the agents condition their strategy on these variables. This makes the regression heavily dependent on the mixture of the agents played and how those agents condition on the control variables. By adding an agent indicator in the regression we reduce, but do not eliminate, this effect.

I regress both methods' models on the semifinals test set, then evaluate each model by calculating the standard error of each finalist's score. Many of the variables in Table 9.4 are not significant. I employ backwards elimination to determine a reduced set of control variables for the strategy-independent and strategy-dependent methods. In backwards elimination, we iteratively remove the variable with the largest p-value above 0.05 until all variables are significant. After variable selection, the strategy-independent method has an average standard error of \$631.87, while the second method has an mean standard error of \$581.71. Adding the agent identity decreased standard error by approximately 8%. Without the adjustments of either model, the mean standard error is \$1,129.34. I chose the strategy-dependent model for our control variates, which decreases standard error by 48%. Table 9.4 gives the coefficients and means selected as the control variates model for the 2009 server. I give the adjusted profit, denoted $\pi^\dagger$, for agent i in game x on the 2009 server as

$$\pi^\dagger(x,\, i) = \pi(x,\, i) - \sum_{\mathrm{q}} \alpha_{\mathrm{q}}(C_{\mathrm{q}}(x,\, i) - \mu_{\mathrm{q}}). \tag{9.2}$$

Control Variable	Mean (μ_{q})	Coefficient (C_{q})	Low (2.5%)	High (97.5%)
C^{cap}	400	0.100	0.092	0.109
N^{ps}	7/9	-1.22	-2.04	-0.39
N^{ms}	7/3	-1.10	-1.57	-0.62
e^{ps}	0.425	36.80	16.11	57.48
γ^{ms}	0.525	15.95	2.26	29.64
γ^{cs}	0.525	17.02	1.85	32.19

Table 9.5: TAC/AA-09 control variates with 95% confidence intervals around the coefficients (in thousands of units).

Table 9.6 compares the raw and adjusted average profit of each finalist. The differences between the two profit measures are relatively small. This is largely attributable to the block-style tournament structure, where effects of the critical distribution capacity on an individual agent's score are marginalized.

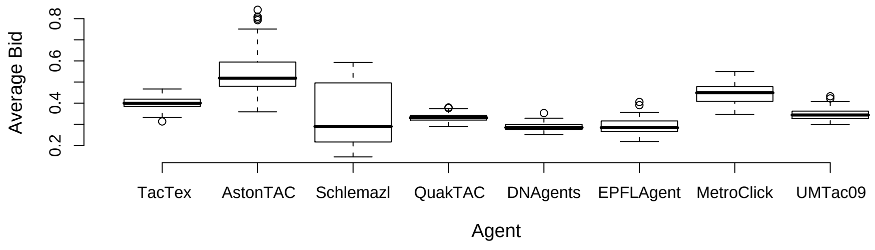
	Profit		Adj. Profit	
Agent	Mean	Std Err	Mean	Std Err
TacTex	79.89	1.25	80.02	0.66
AstonTAC	76.28	1.12	76.52	0.54
Schlemazl	75.41	1.23	75.22	0.87
QuakTAC	74.46	1.05	74.44	0.52
DNAgents	71.78	1.12	72.17	0.53
EPFLAgent	71.69	1.17	72.11	0.54
MetroClick	70.63	0.98	70.83	0.51
UMTac09	66.93	1.10	67.10	0.47

Table 9.6: TAC/AA-09 tournament finalists with raw and adjusted average profit ($K).

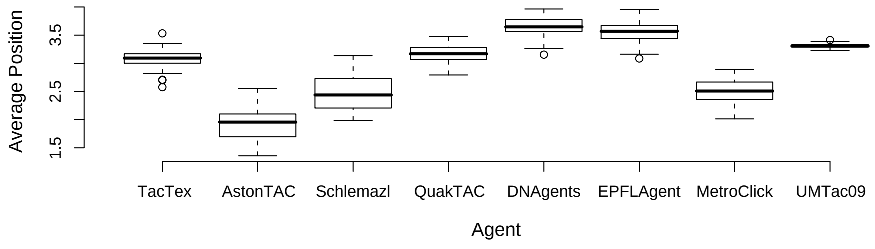
While the order of average profits and thus the overall ranking remains unchanged, the rank within an individual games changed significantly. For instance, TacTex received the highest profit (1^{st} place) in 23 of the 80 finals games, however it received the highest adjusted profit in 36 of the 80 finals games. Similarly, UMTac09 received the lowest profit (8^{th} place) in 18 of the 80 finals games, however it received the lowest adjusted profit in 39 of the 80 finals games.

Managing a Portfolio of Ads

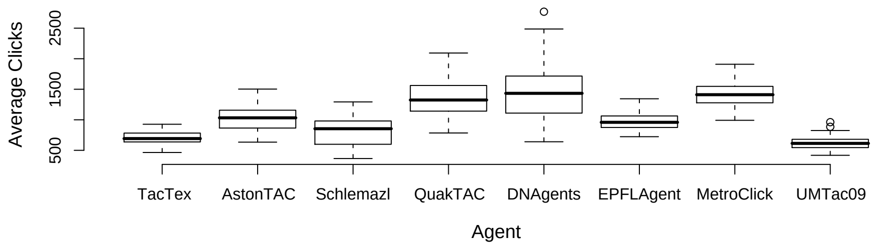
One major component of running an advertising campaign is determining what bid to submit for each ad in a portfolio of ads. The bids determine the position of the ads, which in turn determines the clicks received by the ads. In TAC/AA, the value of a click differs for different query classes, due to the focus and specialization effects. On the other hand, the inventory effect imposes a decreasing marginal conversion rate that is common to all query classes. When agents operate beyond C^{cap}, their conversion probability decreases exponentially. Therefore agents must optimize across a portfolio of ads.



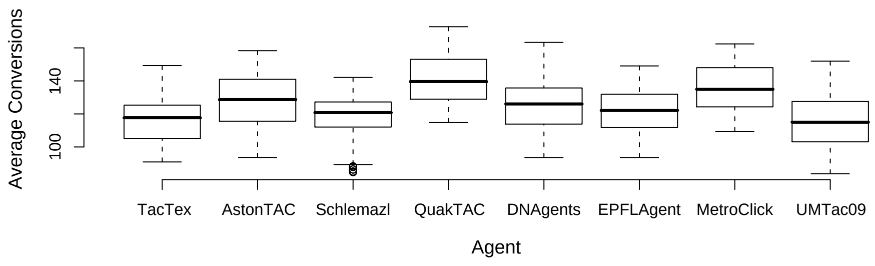
(a) Bid



(b) Position



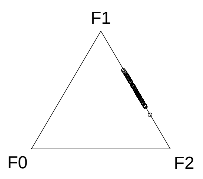
(c) Clicks



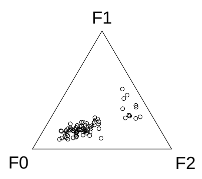
(d) Conversions

Figure 9.5: Average bid, position, clicks, and conversions for each agent in the 2009 TAC/AA finals.

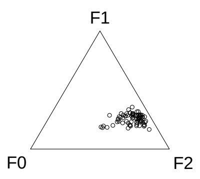
Figure 9.5 shows the average bid, position, number of clicks received, and number of conversions received per game for each agent in the finals. Advertisers influence the distribution over query classes in which they receive clicks by varying bids, ads, and spend limits. This distribution, and the profit associated with each of the clicks, determines the value of the portfolio. Figure 9.6 shows the distribution of clicks over *query types* for each agent. Each point displays the distribution of clicks of an agent for one game.



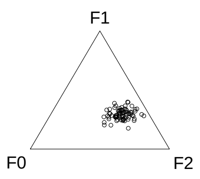
(a) AstonTAC



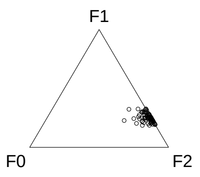
(b) DNAgents



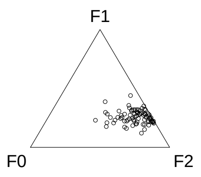
(c) EPFLAgent



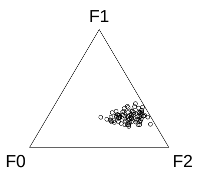
(d) MetroClick



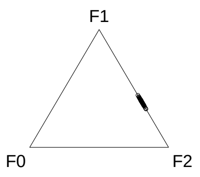
(e) QuakTAC



(f) Schlemazl



(g) TacTex



(h) UMTac09

Figure 9.6: Query type distribution of clicks for each agent in 2009 TAC/AA finals.

Six of the eight agents receive clicks from all three query types. The remaining two, AstonTAC and UMTac09, do not receive any clicks from F0 queries. EPFLAgent, MetroClick, Schlemazl, and TacTex have a relatively large percentage of clicks from each of the query types. While not exclusively limited to F1 and F2 queries, QuakTAC has very few F0 clicks in comparison. On the other hand, DNAgents has a large percentage of F0 clicks, especially after considering that there is a single query class that pertains to the F0 type, while there are 15 that do not.

From the results of the control variates analysis, we know that the parameters corresponding to the manufacturer specialty queries strongly influenced the results of the competition. For the remainder of the section, I investigate how agents condition their observable behavior on these parameters. At the beginning of a simulation each agent is assigned a manufacturer specialty (MS). For each agent, let the queries whose manufacturer information matches the MS be *MS queries* and let the bids that correspond to those queries be *MS bids*. To understand the agents' bids with respect to the manufacturer specialty, I consider each agent's average MS bids over the course of a simulation. Using the sample distributions of MS bids for each pair of agents, I calculate the p-value that the mean is the same using a two-sample Student's t-Test. I use the negative logarithm of the p-values as a dissimilarity measure. Figure 9.7 gives a hierarchical clustering of the agents with the dissimilarity measure. Notice that two highest level clusters split the agents basically according to rank in the tournament, with the exception of MetroClick. The mean MS bids of DNAAgents, EPFLAgent, and UMTac09 all differ significantly from the bids od the remaining five agents. On the other hand, the average MS bids of MetroClick, the second least profitable agent, are more similar to the bids of the top four agents than the remaining agents.

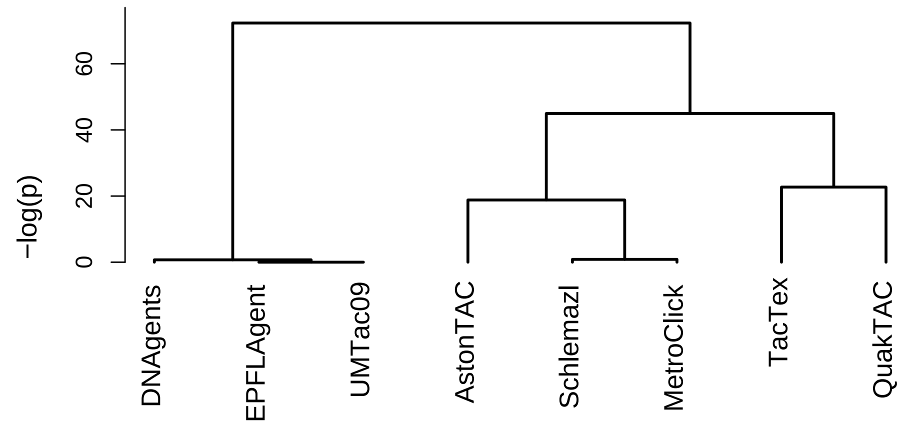


Figure 9.7: Hierarchical cluster of agents' MS bids in the 2009 TAC/AA finals, where $-\log(p)$ is the dissimilarity between two clusters.

I repeat this type of clustering analysis for the average position, number of clicks, and advertising cost of each agent with respect to its respective MS queries. The hierarchical clustering remain qualitatively the same as in Figure 9.7, with the top four agents and MetroClick being separated at the highest level from the remaining three. However, if we cluster according to MS revenue dissimilarity, we see the pattern shown in Figure 9.8.

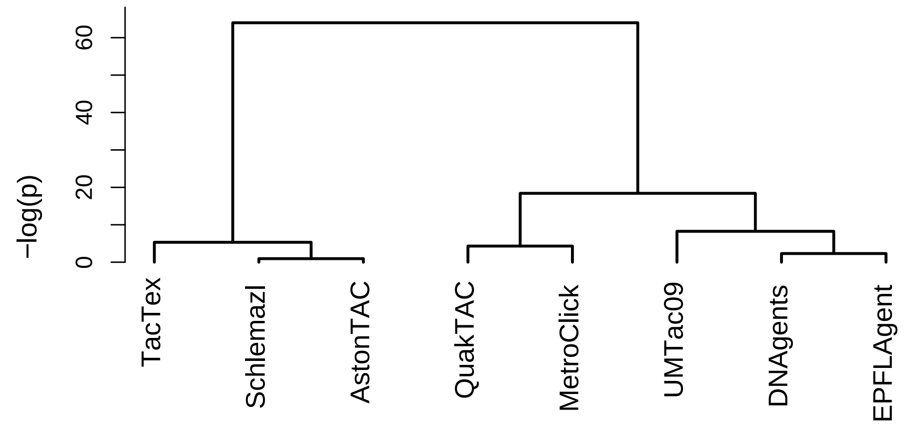


Figure 9.8: Hierarchical cluster of agents' MS revenue in the 2009 TAC/AA finals, where $-\log(p)$ is the dissimilarity between two clusters.

The high-level cluster separates the three most profitable agents from the bottom five. If, for example, we consider the differences in campaign strategy for Schlemazl and MetroClick on MS queries, we see that bids, positions, clicks, and cost are all similar. Therefore the difference of the two agents, at least on MS queries, lies in the revenue. On average, the focus and specialty effects are the same for both agents, therefore it is the inventory effect where the difference is pronounced.

All else being equal, conversions from an agent's manufacturer specialty are the most valuable (50% more than non-MS conversions). Therefore, for a fixed query type and inventory effect, the average value per click is higher for MS queries than non-MS. The other specialty effect, the component specialty, increases the odds of a conversion by 50%, however there is no positive probability, $\pi_l I_d$, such that component specialty bonus increases the conversion rate such that the resulting click value is greater than the click value of the

MS bonus. Therefore, all else being equal, agents should prefer MS clicks over non-MS clicks for the same cost per click.

If we calculate the average fraction of MS clicks each agent receives and cluster using dissimilarity of that fraction, we see two main clusters separating the top three and bottom five agents—similar Figure 9.8. Plotting the adjusted profits against the percentage of MS clicks reveals an even stronger relationship, shown in Figure 9.9. The circles represent data from the top three agents, while the squares represent data from the bottom five agents. The thick black line gives the prediction of a linear model fit to the data. The linear coefficient of the model is significant at a level beyond 0.001, with an adjusted R^2 of 40%.

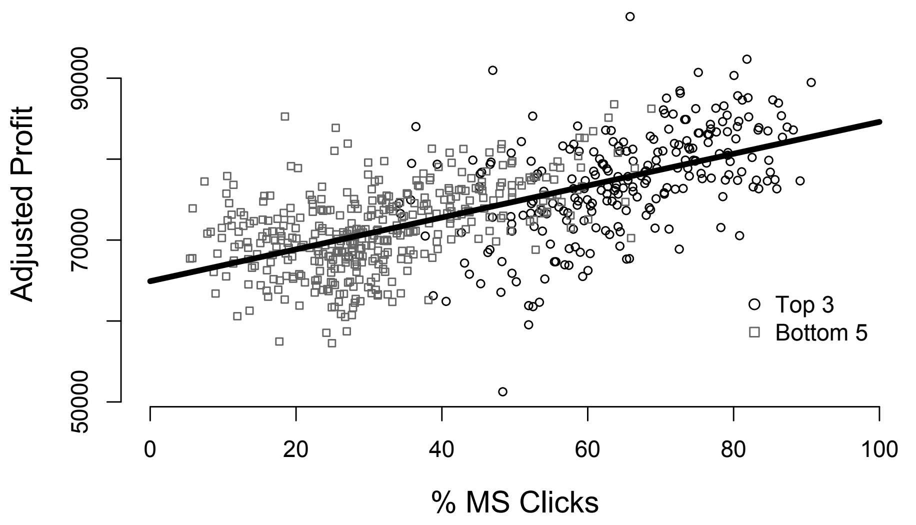


Figure 9.9: Adjusted profits versus the percentage of MS clicks.

The analysis of agents' click distributions reveals a strong relationship between the fraction of clicks associated with MS queries and the adjusted profits. However, the top performing agent (TacTex) had only the third-highest percentage of MS clicks, so other strategic factors play an important role. Consider, for example, the average relative capacity used by an agent:

$$\frac{1}{59}\sum_{d=1}^{59}\sum_{i=d-(W-1)}^{d}\frac{c_i}{C^{cap}}\ .$$

This statistic gives the fraction of the critical distribution capacity is used by the agent each day. Regressing the fraction of MS clicks and the average relative capacity against adjusted profits yields a statistically significant linear coefficient for each variable, however adding the relative capacity only improves the adjusted R^2 of the model by 2%. Thus, while statistically significant, the relative level at which agents keep their recent capacity does not seem to explain the differences in agents' scores. If the difference in relative capacity does not explain the scores, then perhaps how the agents restrict the non-MS clicks plays a role.

One way to restrict clicks on non-MS queries is simply not to be included in the search results corresponding to those queries. For each agent, 12 of the 16 query classes are non-MS classes. For the five least profitable finals' agents, their respective ads appear, on average, in at least 9 of the 12 non-MS query class auctions per day. Of the three most profitable agents, AstonTAC and Schlemazl appear, on average, in at most 4 of the 12 non-MS query class auctions per day.

Another possibility is to apply a binding spend limit on the non-MS queries. All of the three most profitable agents seems to take this approach. When an ad is shown in an auction, each of the top three agents have an average daily binding spent limit on at least seven of the 12 non-MS auctions.

Finally, I return to analyzing the unadjusted profits as a function of capacity and five variables describing agent behavior. Of the five behavior variables, three relate to non-MS behavior, one to MS behavior, and one to the capacity behavior. Table 9.7 lists the variables as well as the cumulative adjusted R^2 after adding each variable to a linear model. The four MS and non-MS variables describe the behavior of agents in terms of how clicks were distributed, while the other two variables express the general degree to which the inventory effect was active. For instance, knowing C^{cap} allows us to explain 60% of the variance, while knowing C^{cap} and the fraction of MS clicks allows us to explain 80%. The coefficient of each variable is significant beyond the 2×10^{-16} level. In total, given the values of the six variables, we can explain 97% of the variance in scores of the finals agents.

Variable	Adjusted R^2
C^{cap}	0.60
Fraction MS clicks	0.80
Average non-MS position	0.85
Average non-MS binding spend limits	0.86
Average used capacity	0.95
Average standard deviation of non-MS spend limits	0.97

Table 9.7: Capacity and significant behavioral variables for predicting advertiser profit.

9.6 Post-Tournament Analysis

After the 2009 tournament, teams were requested to submit their agents to the TAC agent repository. Six of the eight finalists complied, as well as, three of the semifinalists who did not advance to the finals. The players' roles in the simulator are *ex ante* equivalent. With the six strategies, the symmetric eight-player game has a total of 1,287 distinct profiles. I model both the eight-player game and a four-player reduction (Wellman et al., 2005b). As with the TAC/SCM analysis, in the four-player reduction ($\mathrm{AA}{\downarrow_4}$) strategies are assigned to pairs of players rather than individual players, and the payoff to a strategy in a profile is defined as the average payoff to the two players playing this strategy in the original eight-player game.

To date, I have collected results from over 45,000 simulation runs. Each simulation reserves nine CPUs for a period of fifteen minutes—about a third of the computational requirements of a TAC/SCM simulation; one for each agent and one for the TAC/SCM server. As with TAC/SCM, agents can adapt only within a game instance. We scratch simulations where an agent did not communicate with the server for two or more simulation days.

Four-player reduction

For $\mathrm{AA}{\downarrow_4}$, I simulated all 126 profiles, with a minimum of 30 samples per profile. In the resulting empirical game, only TacTex and Schlemazl survive IEDS. There is a mixed-strategy Nash equilibrium where each player chooses TacTex with probability 0.932 and Schlemazl with probability 0.068. Using this equilibrium, I compute the NE regret of each strategy that is listed in Table 9.8, along with its respective max regret. This ranking differs from the tournament ranking in that AstonTAC is ranked lower and EPFLAgent is ranked higher than their respective ranking in the tournament. We can also see that TacTex has a small max regret, implying that it is nearly a dominant strategy in $AA{\downarrow_4}$.

Agent	NE Regret	Max Regret
TacTex	0	1.1
Schlemazl	0	6.9
EPFLAgent	4.0	13.0
AstonTAC	4.2	15.1
QuakTAC	8.9	10.7
MetroClick	32.1	32.8

Table 9.8: TAC/AA-09 tournament $AA{\downarrow_4}$ strategy comparison.

Eight-player empirical game

For the eight-player analysis, I simulated all 495 profiles over five of the six finalist strategies, withholding MetroClick. In this empirical game, there is mixed-strategy Nash equilibrium where each player chooses TacTex with probability 0.90 and AstonTAC with probability 0.10. I tested only deviations from the equilibrium profile to MetroClick, thus reducing the number of required profiles from 1,287 to 503. Using this equilibrium, the NE regret of each strategy is listed in Table 9.9. The ranking under the eight-player model is closer to the tournament ranking than the $AA{\downarrow_4}$ ranking, however EPFLAgent is still ranked higher than its respective ranking in the tournament.

Given the eight-player analysis, we can contrast equilibria in this model with the equilibria in $AA{\downarrow_4}$. If the equilibria in $AA{\downarrow_4}$ are approximate equilibria in the eight-player

Agent	NE Regret
TacTex	0
AstonTAC	0
Schlemazl	0.9
EPFLAgent	1.5
QuakTAC	5.5
MetroClick	22.6

Table 9.9: TAC/AA-09 tournament full strategy comparison.

game, then we can employ the reduction in auction design analysis. Unfortunately, the equilibrium profile in $\text{AA}{\downarrow}_4$ has a regret of \$2.6K in the eight-player game, which is large enough to prefer the eight-player analysis. Interestingly, the eight-player equilibrium mixture has a regret of \$0.5K in $\text{AA}{\downarrow}_4$. Like the tournament analysis of Section 9.5, we find that TacTex, AstonTAC, and Schlemazl are stable strategies. We use these strategies as the basis for optimizing the ad auctions as described in the next section.

9.7 Empirical Auction Design

In this section, I experiment with various ad-auction mechanisms in an attempt to determine a mechanism—a set of allocation and pricing rules—that yields optimal revenue for the publisher. This analysis is consists of two stages: predicting the distribution of play and calculating the revenue under that distribution. Because players condition their strategies on the mechanism, we have to predict the distribution of play for each mechanism. I employ a game-theoretic approach, whereby I use a sample Nash equilibrium of the induced empirical game as a prediction of agent play. Then, I estimate the publisher revenue of the mechanism by calculating the weighted sample mean under the sample equilibrium distribution. In general, this approach—optimizing a mechanism through empirical game-theoretic analysis—is termed *empirical mechanism design*. Empirical mechanism design was first used by Vorobeychik et al. (2006) to analyze day-zero procurement in TAC/SCM and later generalized by Vorobeychik (2008). Like empirical game-theoretic analysis, empirical mechanism design allows us to analyze high-fidelity models. However, there are caveats to the inferences that can be drawn. Principally, the evidence is empirical and the strategy space is limited to a few heuristic strategies. Additionally, I am experimenting with strategies that are designed for one mechanism and using those to determine the revenue under another mechanism.

I solve each mechanism through an eight-player empirical analysis similar to that of Section 9.6. In the worst case, this requires that each of the 1,287 profiles are evaluated. However, based on the analysis from the previous section, I conjecture that the support of a sample Nash equilibrium will contain, at most, TacTex, AstonTAC, and Schlemazl—the top three agents from the tournament. I find that this conjecture is correct. In fact, among all of the tested mechanisms, no sample equilibrium has support that consists of more than two of those strategies. Therefore, we only need to evaluate 9 profiles to determine the equilibrium and 32 profiles to validate it—roughly 3% of profiles in the eight-player, six-strategy game.

In each of the mechanisms, I use generalized second-price (GSP) as the pricing model. I investigate two major components of an ad-auction mechanism: reserve scores and slot scheduling algorithms. For reserve scores, I explore the space of static reserves that are uniform across all auctions. For slot scheduling, I explore algorithms that adjust advertisers' bids in order to spread consumption of spend limits over a simulated day.

I compare auctions according the their expected *publisher revenue*, *advertiser profit*, and *total surplus*. I maximize the publisher revenue and observe the profits for the advertisers during the optimization. The baseline for comparison is the standard 2009 finals mechanism. Table 9.10 gives the publisher revenue, advertiser profits, and total surplus for the profile of agents used in the tournament finals (each of the eight strategies is assigned to a single player) and the eight-player equilibrium profile specified in Section 9.6.

Profile	Publisher Revenue	Advertiser Profit	Total Surplus
Finals	167.31	587.07	754.38
Equilibrium	83.09	688.96	772.05

Table 9.10: TAC/AA baseline comparison [$K].

In equilibrium, the publisher revenue is a small fraction (11%) of the total surplus. This is primarily due to TacTex, which has large support (93.2%) in the equilibrium profile. TacTex tends to keep its bids low and focuses on the MS queries. It also appears that TacTex is implicitly exploiting low reserve scores.

Static reserve prices

Edelman and Schwarz (2007) characterize the optimal auction for the one-shot sponsored search environment proposed by Edelman et al. (2007). They find that in this setting the optimal publisher revenue is achieved by setting a reserve price of

$$\frac{1 - F(v)}{F'(v)},$$

where $F(v)$ is the distribution of IID bidder valuations. The authors use a simulation framework to evaluate the effects of setting reserves. They found that the reserve price matters most when the market is shallow (few advertisers are bidding). In some cases, the optimal reserve price increases publisher revenue more than adding an additional advertiser. This result by Edelman and Schwarz for multi-unit (sponsored search) auctions is in contrast to the findings of Bulow and Klemperer (1996) for single-unit auctions: no setting of the reserve price exceeds the revenue gained through an additional advertiser.

I investigate the effect of reserve scores on publisher revenue in the TAC/AA scenario. TAC/AA is a dynamic, multi-auction generalization of the scenario Edelman and Schwarz (2007) investigated. In the 2009 finals, the publisher set reserve scores for all auctions at the beginning of the simulation. The *regular* and *promoted* reserve scores remained constant throughout the simulation, respectively. The regular reserve score (ρ_{reg}) is drawn uniformly from $[0.02,\ 0.05]$, whereas the promoted reserve score (ρ_{pro}) is drawn uniformly from $[\rho_{\text{reg}},\ 0.4]$. In the first experiment, I constrain $\rho_{\text{reg}} = \rho_{\text{pro}} = \rho$ and vary ρ over the interval $[0.0,\ 1.2]$. At each evaluated reserve score, I select an equilibrium of the game and calculate the publisher revenue and advertiser profits. Figure 9.10 shows equilibrium mixture as a function of reserve score and the publisher revenue, advertiser profits, and total surplus as a function of reserve score.

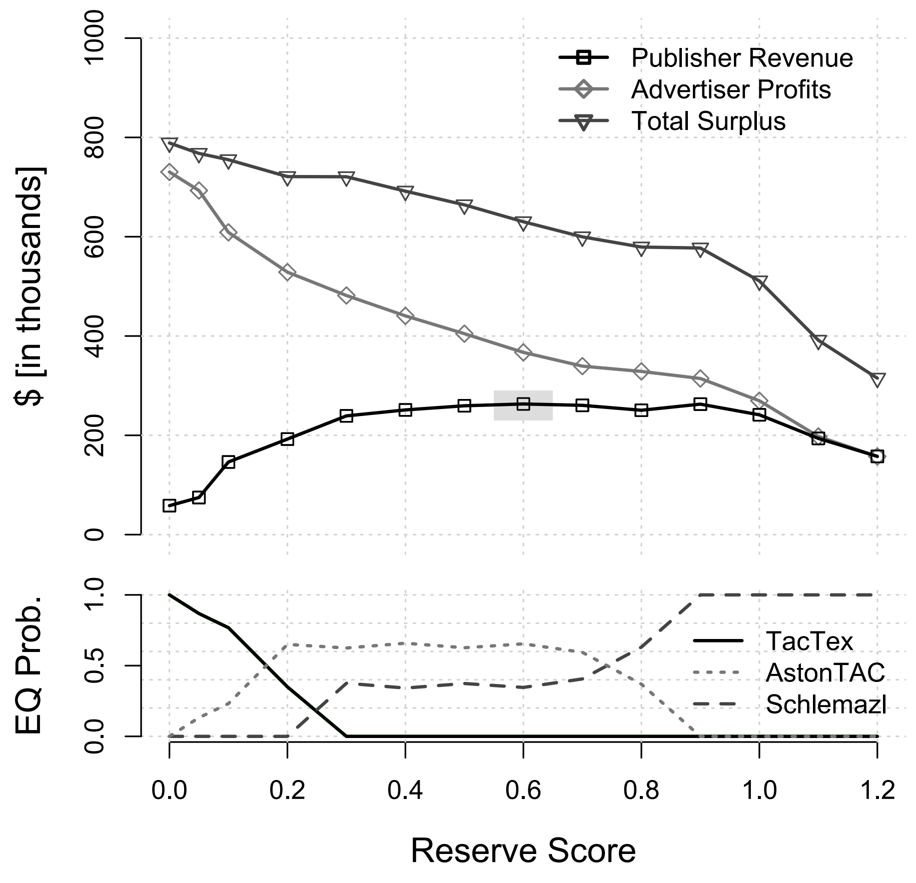


Figure 9.10: Equilibrium mixture, publisher revenue, advertiser profits, and total surplus as a function of reserve score.

The publisher revenue is maximized at a reserve score of 0.60, with a value of approximately \$264K. This is nearly a 400% increase over the publisher surplus without reserve scores ($\rho = 0$). Such a large increase is surprising. For instance, Edelman and Schwarz report an increase of less than 20% in their simulations for eight advertisers; however, their simulations involve a single auction where the eight advertisers have IID valuations. As we have seen in earlier sections, competent advertisers in TAC/AA focus a majority of their spend on MS queries. The expected number of high-value competitors (advertisers whose MS is identical) is $\frac{7}{3}$. While not directly comparable, Edelman and Schwarz (2007) also report a large increase in publisher revenue (approximately 200%) for a similar number of advertisers.

We can also measure the effect of varying numbers of advertisers on publisher revenue. Instead of modifying the simulator to reduce the number of players, we can simply introduce a special type of advertiser called a *zero advertiser*. Zero advertisers submit bids of 0 to the publisher, thus never show ads. Using the optimal reserve setting (0.60), we introduce a varying number of zero advertisers. We compare the publisher revenue for each count of zero advertisers to the revenue when the reserve is set to zero—determining an equilibrium in strategies at each point. Figure 9.11 shows the resulting publisher revenue. The additional revenue garnered from setting reserve scores optimally vastly exceeds the benefit of an additional advertiser. In fact, the publisher prefers its surplus with three advertisers and an optimal reserve score to that with eight advertisers and no reserve score.

Slot Scheduling

When advertisers specify spend limits, the publisher must account for these limits when allocating slots. Advertisers that have consumed their spend limits cannot be charged by the publisher, therefore the publisher does not show the corresponding ads. Determining how to allocate (and price) slots when advertisers specify spend limits is the *slot scheduling problem*.[6]

[6]In practice, this is often called *throttling* or *day parting*.

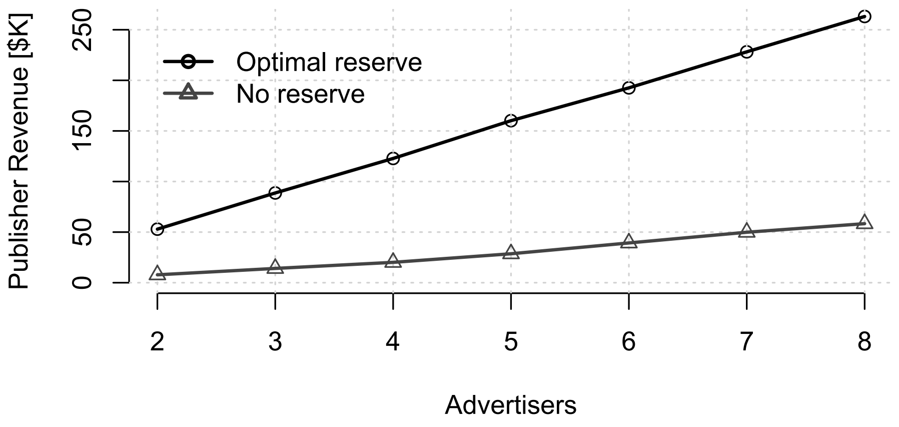


Figure 9.11: Revenue under the optimal reserve and no reserve as a function of advertisers.

Consider a simplified scenario given in Table 9.11. In this scenario, there are two advertisers (*Alice* and *Bob*) and two query classes (*X* and *Y*). For each query, the publisher has one available slot. Assume every time an ad is displayed the user clicks on the ad, each advertiser pays its respective bid when it is allocated a slot, and advertisers are allocated slots until there are no more queries or the advertisers have exhausted their respective spend limits. *How should the publisher allocate slots?*

Advertiser	*X* Bid	*Y* Bid	Spend Limit
Alice	1	0	∞
Bob	$1+\alpha$	$1+\alpha$	$1+\alpha$

Table 9.11: Slot allocation example.

The answer to this question depends on the sequence of queries. For instance, consider a greedy approach where the publisher allocates slots to the highest bidder with an unconsumed spend limit. Using the greedy algorithm, the revenue of the publisher is $2+\alpha$ for the sequence YX. This is the optimal revenue the publisher can achieve for that sequence. However, consider the sequence XY. Using the greedy algorithm, the revenue of the publisher is $1 + \alpha$, whereas the optimal is $2 + \alpha$. By letting α go to zero, we can see that, in the worst case, the revenue of the greedy algorithm achieves at least half of the optimal revenue—thus has a competitive ratio of $\frac{1}{2}$.

Mehta et al. (2007) introduce an algorithm that has a competitive ratio of $1 - 1/e$ for this simplified problem. Let f_i be the fraction of advertiser i's spend limit that has been consumed and $\phi(x) = 1 - e^{x-1}$. The publisher creates adjusted bids $\phi(f_i)b_i$ for allocation and pricing—the slot goes to the advertiser with the highest adjusted bid. Mehta et al. prove that no randomized online algorithm can have a better competitive ratio than $1-1/e$. Lahaie et al. (2007) call this effect "budget smoothing" and, henceforth, I refer to this algorithm as the *budget-smoothing algorithm*. Mahdian et al. (2007) study a modification of the budget-smoothing algorithm that exploits accurate estimates of query frequencies, while maintaining a good worst-case competitive ratio.

I investigate with three different slot allocation algorithms applied to TAC/AA: *greedy*, *budget smoothing*, and *budget smoothing with reserves*. For the budget-smoothing-with-reserves algorithm, the publisher creates adjusted bids of the following form:

$$\phi(f_i)b_i e_i^{\chi} + (1 - \phi(f_i))\rho,$$

where ρ is the reserve score. Thus, instead of tending to zero as f_i approaches one, the with-reserves algorithm tends to the reserve price, ρ/e_i^{χ}. These adjusted bids replace the standard bids in the publisher's allocation and pricing algorithms. The results of the experiment are given in Table 9.12. The greedy algorithm gives substantially better performance, in terms of publisher surplus, than either of the budget smoothing algorithms. The greedy and the budget-smoothing-with-reserves algorithms have comparable total surplus, however the greedy algorithm captures more of the revenue for the publisher with the same setting of reserve scores. The combination of high reserve scores and tending the advertiser bids towards zero seems to doom the standard budget smoothing algorithm, which creates a small surplus for both the publisher and the advertisers.

Algorithm	Publisher Revenue	Advertiser Profits	Total Surplus
Greedy	263.12	367.02	630.14
Budget smoothing	92.19	266.69	358.88
Budget smoothing with reserves	225.44	401.10	626.54

Table 9.12: TAC/AA-09 slot scheduling algorithm comparison [$K].

9.8 Discussion

Internet advertising, in particular ad auctions, is a challenging domain with many interesting problems. Early research in ad auctions characterized equilibrium strategies in abstracted forms of the scenario. Using these abstractions and their corresponding equilibria, researchers have improved the various auction mechanisms used today—both in terms of publisher revenue and total surplus. The TAC/AA scenario is a novel, high-fidelity simulation of an ad auction market sector. It incorporates many features of the domain that are difficult to model analytically, but are nonetheless salient in the design of strategies and mechanisms.

Participants in the 2009 TAC/AA tournament produced 15 viable agent strategies. Of those, three strategies effectively managed to control advertising spend by focusing on high-value query auctions—the primary determinant of success in the tournament. These strategies also fared well in an empirical game-theoretic analysis of the scenario, each in support of an equilibrium under various market parameterizations: TacTex when the reserve score is low, AstonTAC for mid-range reserve scores, and Schlemazl when the reserve is high.

Using these strategies, the auction design experiments demonstrated that reserve scores are a critical component of ad auction design. Even with as few three advertisers participating in the market, the optimal reserve score accounts for more revenue than *eight* advertisers with *no* reserve in TAC/AA. I show that a common slot scheduling algorithm—bid smoothing—can be detrimental to publisher revenue when the reserve scores are set optimally.

Chapter 10
Conclusion

This thesis expands the set of methods and techniques available to modelers when analyzing complex strategic scenarios—in particular, empirical game-theoretic methods and techniques. Below, I summarize my contributions to empirical game theory and the application of these methods and techniques to the analysis of market games.

10.1 Empirical Game-Theoretic Systems

Empirical game-theoretic analysis (EGTA) encompasses the processes by which we construct and analyze an empirical game model from a set of observations. An empirical game-theoretic system unifies the control and analysis of a simulated scenario with the components being operated on—such as a simulator, a set of strategies, and a set of mechanisms. The components of the system (*white boxes with black text*) and the interactions between the components (*arrows*) are illustrated in Figure 10.1. The black arrows represent inputs that select a single element from a set, whereas the gray arrows represent atomic inputs—passing all the data from a component. For instance, the set of strategies—an atomic input—implicitly defines the set of profiles. Selecting a profile to simulate from the set of profiles (Chapter 6 and Chapter 7), however, is a selection input. Given a profile and a mechanism, a simulator generates an observation. Given observations and a set of candidate game models, the system selects a single game model (Chapter 8), then fits the model to the observed data (Chapter 2). The system analyzes the resulting empirical game model (Chapter 3 and Chapter 4) and uses the model to control profile and mechanism selection.

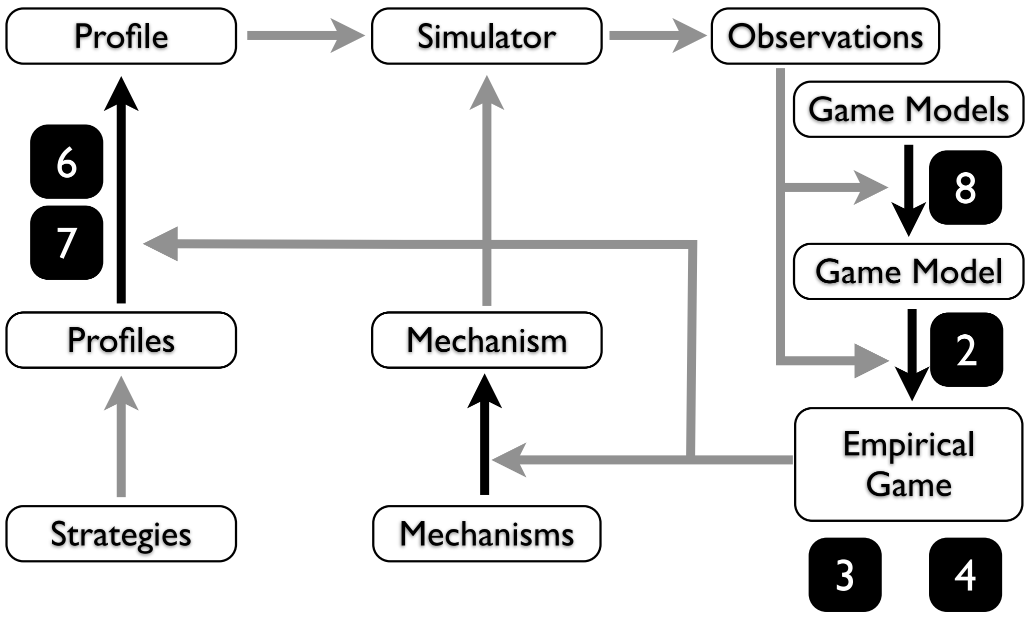


Figure 10.1: Empirical game-theoretic system with relating chapters. *Chapters are given by black boxes with white text; selection inputs are given by black arrows; and atomic inputs are given by gray arrows.*

Estimating model parameters

In Chapter 2, I propose three classes of control variate methods for reducing variance in multiagent simulations. Using these techniques, I demonstrate a significant variance reduction in the TAC/SCM and TAC/AA data sets of Chapters 5 and 9, respectively.

Evaluating strategies

Chapter 3 introduces a novel regret-based methodology for evaluating strategies in an empirical game model. NE regret measures the regret of a strategy with respect to a Nash Equilibrium. Using NE regret, modelers may construct NE-response rankings over strategies. For instance, NE-response rankings may be used to align the incentives of the entrants in research competitions with the goals of the competition designers, where the competition designers desire stable strategies. The methods introduced in Chapter 3 form the basis for strategic analysis in case studies of supply chain management (Chapter 5) and ad auctions (Chapter 9).

Identifying relevant strategies

Many computational problems in game theory, such as finding Nash equilibria, are hard to solve. This limitation forces analysts to limit attention to restricted subsets of the entire strategy space—focusing on strategies that are relevant to analysis. Often this means strategies that support equilibria, or, more generally, strategies that are best responses. In Chapter 4, I investigate formations: strategy sets that are closed under a (correlated) best-response correspondence. I develop algorithms for n-player games that identify formations by modifying an existing family of algorithms for two-player games. I extend these algorithms to apply in cases where the utility function is partially specified, or there is a bound on the size of the restricted profile space. The formation-finding algorithms I introduce outperform existing algorithms from the literature on various classes of games.

Selecting profiles to simulate

For the profile selection problem, we sequentially determine profiles to simulation in order to achieve some goal—for instance, finding a Nash equilibrium. Simulation is costly (often the dominant cost of analysis), therefore we may be limited in the number of simulations we can run. One basic approach is to simulate profiles uniformly until our simulation budget is exhausted. However, if the goal of analysis is to determine a Nash equilibrium, not all profiles are relevant. Of course, we do not know the set of relevant profiles *a priori*. However, during the course of simulation we may determine profiles that are relevant.

I introduce novel policies, as well as extend existing policies, for sequentially determining profiles to simulate, when constrained by a budget for simulation. I consider two formulations of the profile selection problem: the revealed-payoff model and the noisy-payoff model. In Chapter 6, I investigate policies that search for low-regret profiles (approximate equilibria). For the revealed-payoff model, I compare minimum-regret-first search algorithm (MRFS) to the tabu best-response algorithm (TABU). TABU is the only existing algorithms from the literature that has formally been analyzed. MRFS was informally used in analysis of TAC/SCM. I find that MRFS is superior to TABU in many games of interest. For the noisy-payoff model, I propose the information-gain search algorithm (IGS). This algorithm outperforms the existing expected-confirmational-value-of-information algorithm from the literature (ECVI). The MRFS and IGS algorithms for the revealed-payoff and noisy-payoff models of observation, respectively, significantly reduce the number of observations required to identify an approximate equilibrium when compared to the other algorithms from the literature.

Schvartzman and Wellman (2009a) describe a special case of the profile selection problem, called the strategy exploration problem. Unlike profile selection policies such as MRFS and IGS, strategy exploration policies sequentially determine restricted games (sets of profiles) to evaluate. In Chapter 7, I propose a novel formation-based policy, MEMT, for the revealed-payoff model. In an experimental study, MEMT performed as well as or better than to the best existing policies in each case.

Selecting an empirical game model

In empirical game-theoretic systems, the empirical game that models the strategic aspects of the scenario is constructed from empirical observation. This presents a tradeoff between granularity and statistical confidence. For instance, collecting a large amount of data about each profile improves confidence, but restricts the range of profiles that can be explored. I introduce a flexible approach, where we may construct multiple game-theoretic models from the same set of observations. The models may vary in their expressiveness—ability to represent a given utility function. I propose a family of algorithms in Chapter 8 that select an empirical game model from a set of candidates. The algorithms use the concept of generalization risk—treating each empirical game model as a predictor of payoffs—to compare model classes. The model selection algorithms are sufficiently general such that many compact game representations can be reasoned over—for instance, factored games and graphical games. I demonstrate the efficacy of this approach by analyzing equivalent strategy models on the TAC/SCM scenario.

10.2 Market Games

I employ the developed empirical game-theoretic techniques in two case studies of market scenarios. In the first market scenario, I analyze the supply chain management strategies through the perspective of a strategy designer. In the second, I create a simulated ad-auction market and optimize the auction mechanism using strategies provided by external research teams.

Supply chain management

Through a case study of the TAC/SCM market scenario (Chapter 5), I find evidence of agent progress over subsequent tournaments, an important measure of the success for the competition. In addition, I describe the regret-based development approach taken by the DeepMaize team. I find that early-game procurement—the policy for selecting quantities and order dates for purchases in the early part of the game—is a significant factor in the regret (stability) of a strategy. When set optimally with respect to the tested policies, each player choosing DeepMaize 2008 is a pure-strategy Nash equilibrium, and the strategy set consisting of only DeepMaize is the sole primitive formation.

In part, the analysis of the DeepMaize 2008 candidate strategies isolated the effect of early-game procurement policies. We could have, for instance, selected a policy that was a best response to the 2007 finals agents. However, this policy would have likely been too aggressive. The evidence from the 2008 tournament scores suggests that many teams selected policies that were too aggressive for their respective agent. Instead we chose an equilibrium policy, based on empirical game-theoretic analysis. Thus, at least anecdotally, I find evidence that this empirical game-theoretic approach to parameter selection contributed to championship finishes in the 2008 and 2009 tournaments.

Ad auctions

I present a case study of ad auctions in Chapter 9. With my co-developers, I design a scenario (TAC/AA) capturing key strategic issues in the Internet ad-auction domain for the first time. TAC/AA debuted in summer 2009, with fifteen research groups participating. I isolate strategic behavior that separates the top three agents from the remaining agents. Using these agents as a heuristic strategy set, I perform a strategic analysis of the scenario and find that the top three agents from the tournament are the three most strategically stable.

Finally, I investigate various auction mechanisms. In particular, I vary the reserve score and determine a Nash equilibrium over tournament strategies for each setting. In each case, the set of profiles evaluated is a fraction (roughly 3%) of the total space. By exploiting small supports, I was able to finely sample the reserve score. I demonstrate that, from the publisher's perspective, the reserve score is a principal determinant of publisher revenue in the TAC/AA scenario and, thus, finding the optimal reserve score is a core problem for the publisher.

Bibliography

V. Abhishek. Keyword generation for search engine advertising using semantic similarity between terms. In *WWW-07 Workshop on Sponsored Search Auctions*, 2007.

Z. Abrams and A. Gosh. Auctions with revenue guarantees for sponsored search. In *Workshop on Internet and Network Economics*, pages 143–154, 2007.

Z. Abrams and M. Schwarz. Ad auction design and user experience. *Applied Economics Research Bulletin*, Special Issue I(Auctions):98–105, March 2008.

S. Acharya, P. Krishnamurthy, K. Deshpande, T. Yan, and C.-C. Chang. A simulation framework for evaluating designs for sponsored search markets. *WWW-07 Workshop on Sponsored Search Auctions*, 2007.

G. Aggarwal, A. Goel, and R. Motwani. Truthful auctions for pricing search keywords. In *Seventh ACM Conference on Electronic Commerce*, pages 1–7, Ann Arbor, June 2006.

G. Aggarwal, J. Feldman, S. Muthukrishnan, and M. Pal. Sponsored search auctions with Markovian users. In *Fourth International Workshop on Internet and Network Economics*, LNCS 5385, pages 621–628, Shanghai, 2008.

G. Aggarwal, S. Muthukrishnan, D. Pal, and M. Pal. General auction mechanism for search advertising. In *Eighteenth International World Wide Web Conference*, pages 241–250, Madrid, 2009.

O. Armantier, J.-P. Florens, and J.-F. Richard. Approximation of Bayesian Nash equilibrium. *Journal of Applied Econometrics*, 23(7):965–981, 2008.

R. Arunachalam and N. M. Sadeh. The supply chain trading agent competition. *Electronic Commerce Research and Applications*, 4:63–81, 2005.

S. Athey and G. Ellison. Position auctions with consumer search. Working paper, UCLA Department of Economics, October 2007.

K. Bartz, V. Murthi, and S. Sebastian. Logistic regression and collaborative filtering for sponsored search term recommendation. In *ACM EC-06 Workshop on Sponsored Search Auctions*, Ann Arbor, 2006.

K. Basu and J. W. Weibull. Strategy subsets closed under rational behavior. *Economic Letters*, 36(2):141–146, 1991.

M. Benisch, A. Greenwald, I. Grypari, R. Lederman, V. Naroditskiy, and M. C. Tschantz. Botticelli: A supply chain management agent. In *Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1174–1181, July 2004.

M. Benisch, G. B. Davis, and T. Sandholm. Algorithms for rationalizability and CURB sets. In *Twenty-First National Conference on Artificial Intelligence*, pages 598–604, Boston, July 2006.

M. Benisch, J. Andrews, A. Sardinha, R. Ravichandran, and N. Sadeh. CMieux: Adaptive strategies for competitive supply chain trading. *Electronic Commerce Research and Applications*, 8(2):78–90, 2009.

F. Bergeron and L. Raymond. The advantages of electronic data interchange. *ACM SIGMIS Database*, 23(4):19–31, 1992.

B. D. Bernheim. Rationalizable strategic behavior. *Econometrica*, 52(4):1007–1028, 1984.

T. Börgers, I. J. Cox, M. Pesendorfer, and V. Petricek. Equilibrium bids in auctions of sponsored links: Theory and evidence. Working paper, September 2007.

M. Bowling, M. Johanson, N. Burch, and D. Szafron. Strategy evaluation in extensive games with importance sampling. In *Twenty-Fifth International Conference on Machine Learning*, pages 72–79, 2008.

F. Brandt, M. Brill, F. Fischer, and P. Harrenstein. Computational aspects of Shapley's saddles. In *Eighth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 209–216, 2009.

J. Bulow and P. Klemperer. Auctions versus negotiations. *American Economic Review*, 86 (1):180–194, March 1996.

M. Cary, A. Das, B. Edelman, I. Goitis, K. Heimerl, A. R. Karlin, C. Mathieu, and M. Schwartz. Greedy bidding strategies for keyword auctions. In *Eighth ACM Conference on Electronic Commerce*, pages 262 – 271, San Diego, 2007.

M. Cary, A. Das, B. Edelman, K. H. Ioannis Giotis, A. R. Karlin, C. Mathieu, and M. Schwarz. On best-response bidding in GSP auctions. Working Paper 13788, National Bureau of Economic Research, February 2008.

X. Chen and X. Deng. Settling the complexity of 2-player Nash-equilibrium. In *Forty-Seventh Annual IEEE Symposium on Foundations of Computer Science*, pages 261–272, 2006.

Y. Chen and C. He. Paid placement: Advertising and search on the Internet. Working Paper No. 06-02, NET Institute, September 2006.

Y. Chen, G.-R. Xue, and Y. Yu. Advertising keyword suggestion based on concept hierarchy. In *First ACM International Conference on Web Search and Data Mining*, pages 251–260, Stanford, February 2008.

S.-F. Cheng and M. P. Wellman. Iterated weaker than-weak dominance. In *Twentieth International Joint Conference on Artificial Intelligence*, pages 1233–1238, 2007.

J. Collins and W. Ketter. An experiment management framework for TAC SCM agent evaluation. In *IJCAI-09 Workshop on Trading Agent Design and Analysis*, Pasadena, July 2009.

J. Collins, R. Arunachalam, N. Sadeh, J. Eriksson, N. Finne, and S. Janson. The Supply Chain Management Game for the 2005 Trading Agent Competition. Technical Report CMU-ISRI-04-139, Carnegie Mellon University, 2004.

J. Collins, W. Ketter, M. Gini, and A. Agovic. Software architecture of the MinneTAC supply-chain trading agent. Technical Report 07-006, University of Minnesota, Dept of Computer Science and Engineering, Minneapolis, 2007.

V. Conitzer and T. Sandholm. A generalized strategy eliminability criterion and computational methods for applying it. In *Twentieth National Conference on Artificial Intelligence*, volume 2, pages 483–488, 2005.

N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *First ACM Conference on Web Search and Data Mining*, Stanford, 2008.

A. Das, I. Goitis, A. R. Karlin, and C. Mathieu. On the effects of competing advertisements in keyword auctions. Unpublished Manuscript, May 2008.

C. Daskalakis and C. H. Papadimitriou. Three-player games are hard. *Electronic Colloquium on Computational Complexity*, 12(139), 2005.

C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *Electronic Colloquium on Computational Complexity*, 12(139), 2005.

Q. Duong, M. P. Wellman, and S. Singh. Knowledge combination in graphical multiagent models. In *Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 153–160, July 2008.

B. Edelman and M. Schwarz. Optimal auction design in a multi-unit environment: The case of sponsored search auctions. In *Eighth ACM Conference on Electronic Commerce*, 2007.

B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97:242–259, 2007.

I. Erev and A. E. Roth. Predicting how people play games: Reinforcement learning in experimental games with unique, mixed-strategy equilibria. *American Economic Review*, 88:848–881, 1998.

J. Eriksson, N. Finne, and S. Janson. Evolution of a supply chain management game for the trading agent competition. *AI Communications*, 19:1–12, 2006.

E. Even-Dar, J. Feldman, Y. Mansour, and S. Muthukrishnan. Position auctions with bidder-specific minimum prices. In *Internet and Network Economics*, LNCS 5385, pages 577–584. Springer, 2008.

E. Even-Dar, Y. Mansour, V. S. Mirrokni, S. Muthukrishnan, and U. Nadav. Bid optimization for broad match ad auctions. In *Eighteenth International World Wide Web Conference*, pages 231–240, Madrid, April 2009.

D. C. Fain and J. O. Pedersen. Sponsored search: A brief history. *Bulletin of the American Society for Information Science and Technology*, 32(2):12–13, 2006.

J. Feldman and S. Muthukrishnan. Algorithmic methods for sponsored search advertising. In Z. Liu and C. H. Xia, editors, *Performance Modeling and Engineering*, pages 91–124. Springer, 2008.

S. G. Ficici, D. C. Parkes, and A. Pfeffer. Learning and solving many-player games through a cluster-based representation. In *Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 187–195, Helsinki, 2008.

S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias / variance dilemma. *Neural Computation*, 4(1):1–58, January 1992.

A. Ghose and S. Yang. Analyzing search engine advertising: Firm behavior and cross-selling in electronic markets. In *17th International World Wide Web Conference*, pages 219–226, Beijing, April 2008.

M. Gini. To compete or not compete? Ingredients for a successful competition. In *IJCAI-09 Workshop on Competitions in Artificial Intelligence and Robotics*, July 2009.

L. C. Giunipero, R. E. Hooker, S. Joseph-Matthews, T. E. Yoon, and S. Brudvig. A decade of SCM literature: Past, present and future implications. *Journal of Supply Chain Management*, 44(4):66–86, 2008.

G. Goel and A. Mehta. Online budgeted matching in random input models with applications to adwords. In *Nineteenth ACM-SIAM Symposium on Discrete Algorithms*, pages 982–991, San Francisco, 2008.

A. Gunawardana and C. Meek. Aggregators and contextual effects in search ad markets. In *WWW-08 Workshop on Targeting and Ranking for Online Advertising*, 2008.

Q. Guo, E. Agichtein, C. L. A. Clarke, and A. Ashkan. In the mood to click? Towards inferring searcher receptiveness to search advertising. In *IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technologies*, pages 319–324, September 2009.

J. Y. Halpern and R. Pass. Iterated regret minimization: A new solution concept. In *Twenty-First International Joint Conference on Artificial Intelligence*, pages 153–158, 2009.

J. M. Hammersley and D. C. Handscomb. *Monte Carlo Methods*. Methuen & Co. Ltd., London, 1964.

J. C. Harsanyi. Games with incomplete information played by "Bayesian" players. Part I. The basic model. *Management Science*, 14:159–82, 1967.

J. C. Harsanyi and R. Selten. *A General Theory of Equilibrium Selection in Games*. MIT Press, June 1988.

M. He, A. Rogers, X. Luo, and N. R. Jennings. Designing a successful trading agent for supply chain management. In *Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1159–1166, Hakodate, May 2006.

B. J. Jansen, D. L. Booth, and A. Spink. Determining the informational, navigational, and transactional intent of Web queries. *Information Processing and Management*, 44: 1251–1266, 2008.

P. R. Jordan, C. Kiekintveld, and M. P. Wellman. Empirical game-theoretic analysis of the TAC supply chain game. *Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 1188–1195, May 2007.

P. R. Jordan, B. Cassell, L. F. Callender, and M. P. Wellman. The trading agent competition ad auctions game. Technical report, University of Michigan, 2009. URL `http://aa.tradingagents.org`.

E. Kalai and D. Samet. Persistent equilibria in strategic games. *International Journal of Game Theory*, 14(3):129–144, 1984.

D. Kempe and M. Mahdian. A cascade model for externalities in sponsored search. In *Fourth International Workshop on Internet and Network Economics*, pages 585–596, Shanghai, 2008.

W. Ketter, J. Collins, and M. Gini. A survey of agent designs for TAC SCM. *AAAI-08 Workshop on Trading Agent Design and Analysis (TADA)*, 2008.

C. Kiekintveld, J. Miller, P. R. Jordan, and M. P. Wellman. Controlling a supply chain agent using value-based decomposition. In *Seventh ACM Conference on Electronic Commerce*, pages 208–217, Ann Arbor, 2006.

C. Kiekintveld, J. Miller, P. R. Jordan, L. F. Callender, and M. P. Wellman. Forecasting market prices in a supply chain game. *Electronic Commerce Research and Applications*, 8(2):63–77, 2009.

B. Kitts and B. Leblanc. Optimal bidding on keyword auctions. *Electronic Markets*, 14(3): 186–201, 2004.

T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45(3):385–482, 2003.

S. D. Kominers. Dynamic position auctions with consumer search. Working paper, Harvard University, November 2008.

I. Kontogounis, K. C. Chatzidimitriou, A. L. Symeonidis, and P. A. Mitkas. A robust agent design for dynamic SCM environments. In *Fourth Hellenic Conference on Artificial Intelligence*, Heraklion, Greece, 2006.

S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

S. Lahaie and D. M. Pennock. Revenue analysis of a family of ranking rules for keyword auctions. In *Eighth ACM Conference on Electronic Commerce*, pages 50–56, San Diego, 2007.

S. Lahaie, D. M. Pennock, A. Saberi, and R. V. Vohra. Sponsored search auctions. In N. Nisan, T. Roughgarden, Éva Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*, chapter 28, pages 699–716. Cambridge University Press, 2007.

S. S. Lavenberg and P. D. Welch. A perspective on the use of control variables to increase the efficiency of Monte Carlo simulations. *Management Science*, 27(3):322–335, 1981.

S. S. Lavenberg, T. L. Moeller, and P. D. Welch. Statistical results on control variables with applications to queueing network simulation. *Operations Research*, 30(1):182–202, 1982.

P. L'Ecuyer. Efficiency improvement and variance reduction. In *Winter Simulation Conference*, pages 122–132. IEEE Press, December 1994.

M. Mahdian and A. Saberi. Multi-unit auctions with unknown supply. In *Seventh ACM Conference on Electronic Commerce*, pages 243–249, Ann Arbor, 2006.

M. Mahdian, H. Nazerzadeh, and A. Saberi. Allocating online advertisement space with unreliable estimates. In *Eighth ACM Conference on Electronic Commerce*, pages 288–294, San Diego, June 2007.

A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. Adwords and generalized on-line matching. *Journal of the ACM*, 54(5), 2007.

N. Metropolis. The beginning of the Monte Carlo method. *Los Alamos Science*, (15): 125–130, 1987.

N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

D. Monderer and L. S. Shapley. Potential games. *Games and Economic Behavior*, 14(44): 124–143, 1996.

S. Muthukrishnan. Internet ad auctions: Insights and directions. In *Thirty-Fifth International Colloquium on Automata, Languages, and Programing (ICALP)*, LNCS 5125, pages 14–23, Reykjavik, July 2008. Springer.

S. Muthukrishnan, M. Pal, and Z. Svitkina. Stochastic models for budget optimization in search-based advertising. In *Internet and Network Economics*, Lecture Notes in Computer Science, pages 131–142. Springer, 2007.

J. Nash. Non-cooperative games. *The Annals of Mathematics*, 54(2):286–295, 1951.

J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

B. L. Nelson. An illustration of the sample space definition of simulation and variance reduction. *Transactions of the Society for Computer Simulation*, 2(3):237–247, 1985.

J. Niu, K. Cai, S. Parsons, E. Gerding, and P. McBurney. Characterizing effective auction mechanisms: Insights from the 2007 TAC market design competition. In *Seventh International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 1079–1086, Estoril, Portugal, 2008.

E. Nudelman, J. Wortman, Y. Shoham, and K. Leyton-Brown. Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 880–887, 2005.

M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

D. Pardoe and P. Stone. TacTex-05: A champion supply chain management agent. In *Twenty-First National Conference on Artificial Intelligence*, pages 1489–1494, Boston, 2006.

D. Pardoe and P. Stone. An autonomous agent for supply chain management. In G. Adomavicius and A. Gupta, editors, *Handbooks in Information Systems Series: Business Computing*. Elsevier, 2007.

D. Pardoe and P. Stone. The 2007 TAC SCM prediction challenge. In *AAAI-08 Workshop on Trading Agent Design and Analysis*, July 2008.

D. G. Pearce. Rationalizable strategic behavior and the problem of perfection. *Econometrica*, 52(4):1029–1050, July 1984.

V. Podobnik, A. Petric, and G. Jezic. The CrocodileAgent: Research for efficient agent-based cross-enterprise processes. In *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, pages 752–762, 2006.

R. Porter, E. Nudelman, and Y. Shoham. Simple search methods for finding a Nash equilibrium. *Games and Economic Behavior*, 63(2):642–662, July 2008.

M. Powell. Direct search algorithms for optimization calculations. *Acta Numerica*, 7: 287–336, 1998.

D. M. Reeves. *Generating Trading Agent Strategies: Analytic and Empirical Methods for Infinite and Large Games*. PhD thesis, University of Michigan, 2005.

D. M. Reeves, M. P. Wellman, J. K. MacKie-Mason, and A. Osepayshvili. Exploring bidding strategies for market-based scheduling. *Decision Support Systems*, 39:67–85, 2005.

R. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973.

S. M. Ross. *Simulation*. Academic Press, 3rd edition, 2001.

P. Rusmevichientong and D. P. Williamson. An adaptive algorithm for selecting profitable keywords for search-based advertising services. In *Seventh ACM Conference on Electronic Commerce*, pages 260–269, Ann Arbor, 2006.

S. Russell and E. Wefald. Principles of metareasoning. *Artificial Intelligence*, 49:361–395, 1991.

L. J. Savage. *The Foundations of Statistics*. John Wiley and Sons, New York, 1954.

L. J. Schvartzman and M. P. Wellman. Exploring large strategy spaces in empirical game modeling. In *Eleventh International Workshop on Agent-Mediated Electronic Commerce*, May 2009a.

L. J. Schvartzman and M. P. Wellman. Stronger CDA strategies through empirical game-theoretic analysis and reinforcement learning. In *Eighth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Budapest, May 2009b.

E. Sodomka, J. Collins, and M. Gini. Efficient statistical methods for evaluating trading agent performance. In *Twenty-Second AAAI Conference on Artificial Intelligence*, pages 770–775, Vancouver, July 2007.

M. Stan, B. Stan, and A. M. Florea. A dynamic strategy agent for supply chain management. In *Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, pages 227–232, 2006.

M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, 36(1):111–147, 1974.

A. Sureka and P. R. Wurman. Using tabu best-response search to find pure strategy Nash equilibria in normal form games. In *Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1023–1029, Utrecht, 2005.

H. R. Varian. Position auctions. *International Journal of Industrial Organization*, 25: 1163–1178, 2007.

J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, New Jersey, 1944.

M. Voorneveld. Preparation. *Games and Economic Behavior*, 48:403–414, 2004.

M. Voorneveld. Persistent retracts and preparation. *Games and Economic Behavior*, 51: 228–232, 2005.

Y. Vorobeychik. *Mechanism Design and Analysis Using Simulation-Based Game Models*. PhD thesis, University of Michigan, 2008.

Y. Vorobeychik and D. M. Reeves. Equilibrium analysis of dynamic bidding in sponsored search auctions. *International Journal of Electronic Business*, 6(2):172–193, 2008.

Y. Vorobeychik and M. P. Wellman. Mechanism design based on beliefs about responsive play (position paper). In *ACM EC-06 Workshop on Alternative Solution Concepts for Mechanism Design*, Ann Arbor, 2006.

Y. Vorobeychik, C. Kiekintveld, and M. P. Wellman. Empirical mechanism design: Methods, with application to a supply chain scenario. *Seventh ACM Conference on Electronic Commerce*, pages 306–315, 2006.

W. Walsh, D. Parkes, and R. Das. Choosing samples to compute heuristic-strategy Nash equilibrium. In *Fifth Workshop on Agent-Mediated Electronic Commerce*, 2003.

W. E. Walsh, R. Das, G. Tesauro, and J. O. Kephart. Analyzing complex strategic interactions in multi-agent systems. In *AAAI-02 Workshop on Game-Theoretic and Decision-Theoretic Agents*, Edmonton, 2002.

M. P. Wellman. Methods for empirical game-theoretic analysis. In *Twenty-First National Conference on Artificial Intelligence*, pages 1152–1155, Boston, MA, 2006.

M. P. Wellman, J. Estelle, S. Singh, Y. Vorobeychik, C. Kiekintveld, and V. Soni. Strategic interactions in a supply chain game. *Computational Intelligence*, 21(1):1–26, February 2005a.

M. P. Wellman, D. M. Reeves, K. M. Lochner, S.-F. Chen, and R. Suri. Approximate strategic reasoning through hierarchical reduction of large symmetric games. In *Twentieth National Conference on Artificial Intelligence*, pages 502–508, Pittsburgh, 2005b.

M. P. Wellman, A. Greenwald, and P. Stone. *Autonomous Bidding Agents: Strategies and Lessons from the Trading Agent Competition*. MIT Press, 2007.

J. R. Wilson. Variance reduction techniques for digital simulation. *American Journal of Mathematical and Management Sciences*, 4(3):277–312, 1984.

J. Wortman, Y. Vorobeychik, L. Li, and J. Langford. Maintaining equilibria during exploration in sponsored search auctions. In *Workshop on Internet and Network Economics*, pages 119–130, 2007.

Y. Zhou and R. Lukose. Vindictive bidding in keyword auctions. In *Ninth International Conference on Electronic Commerce*, pages 141–146, Minneapolis, 2007.

Y. Zhou and V. Naroditskiy. An algorithm for stochastic multiple-choice knapsack problem and keywords bidding. In *Seventeenth International World Wide Web Conference*, pages 1175–1176, Beijing, April 2008.

Y. Zhou, D. Chakrabarty, and R. Lukose. Budget constrained bidding in keyword auctions and online knapsack problems. In *Internet and Network Economics*, Lecture Notes in Computer Science, pages 566–576. Springer, 2008.

M. Zinkevich, M. Bowling, N. Bard, M. Kan, and D. Billings. Optimal unbiased estimators for evaluating agent performance. In *Twenty-First National Conference on Artificial Intelligence*, pages 573–578, 2006.