# A Framework for Modeling Population Strategies by Depth of Reasoning

Michael Wunder
Rutgers University
mwunder@cs.rutgers.edu

Michael Kaisers
Maastricht University
michael.kaisers@maastrichtuniversity.nl

John Robert Yaros
Rutgers University
yaros@cs.rutgers.edu

Michael Littman
Rutgers University
mlittman@cs.rutgers.edu

## ABSTRACT

This article presents a population-based cognitive hierarchy model that can be used to estimate the reasoning depth and sophistication of a collection of opponents' strategies from observed behavior in repeated games. This framework provides a compact representation of a distribution of complicated strategies by reducing them to a small number of parameters. This estimated population model can be then used to compute a best response to the observed distribution over these parameters. As such, it provides a basis for building improved strategies given a history of observations of the community of agents. Results show that this model predicts and explains the winning strategies in the recent 2011 Lemonade Stand Game competition, where eight algorithms were pitted against each other. The Lemonade Stand Game is a three-player game with simple rules that includes both cooperative and competitive elements. Despite its apparent simplicity, the fact that success depends crucially on what other players do gives rise to complex interaction patterns, which our new framework captures well.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

## Keywords

Iterated Reasoning, Learning in populations, Multiagent Learning

## 1. INTRODUCTION

The essential problem in multiagent learning has been to apply lessons and techniques from the well-developed field of machine learning to dynamic environments where other decision makers are present. In addition to the obvious challenges of non-stationarity and adversarial adaptation, we might also consider worlds where agents remain anonymous, for one reason or another, such that we may not know at any given moment who we are playing against. This objective, which can be found in such diverse settings as financial markets and politics, raises questions like how to encode a group of complicated strategies compactly so that we may know how to preempt them when a new situation arises. To put the problem in a machine-learning context, we would like to assemble the right feature set to support learning.

Game theory has been built around the concept of Nash equilibria. On the one hand, some games have no unique equilibrium and in many others it is not efficiently computable. On the other hand, if an agent's success primarily depends on others' actions, equilibria may be ubiquitous and completely lose their interpretation as a solution concept. As a result, researchers have developed an alternative theory which has subsequently been proven relevant to human behavior empirically by a wide array of behavioral experiments. This new class of models goes by many names, but the basic idea is that strategies can be classified according to a cognitive hierarchy (CH) with non-reasoning behavior at the bottom and progressively more strategic reasoning at higher levels [4, 5]. There are several hypothesis why this phenomenon occurs. Perhaps people wish to put a minimal amount of effort to the task, or are limited in capabilities. Maybe everyday experience gives them a reasonably good model of what to expect, and this model leads to the behavior. The important thing to note is that this finding appears to be universal, across different cultures and personal backgrounds and over many different games [3].

This paper utilizes and extends an approach that has been applied successfully to behavioral game theory data in the single-shot case [13]. The contribution of this line of previous work was to evalutate a range of models by determining how well they predicted unseen behaviors in matrix-game experiments played by people. This work parallels and builds upon the prior frameworks in several ways but diverges in others. First, we are aligned in our goal of predicting behavior given a data set of obervations, rather than merely explaining the observed behavior. Another shared focus is on the quantal level-$k$ model which has the same properties we would expect to see in our case. While we are primarily interested in repeated settings, unlike the earlier work, there are still similarities to the single-shot case that motivate this approach. For instance, the initial action selection problem can be viewed as a single-shot game given that there is no history, especially if the start to a game significantly influences the course of the succeeding actions. We propose that certain games, like the example case we examine in detail, consist of a series of state-based decisions that closely resemble single-shot games. One of our main contributions is

**Appears in:** *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

to develop novel statistical methods that can be applied to sequential decisions. While we are studying games played by software agents rather than humans, we maintain that reasoning agents - both programs and humans - are driven by the same behavioral interaction patterns and conclusions of our experiments transfer to human behavior. One benefit to analyzing simulations is the ease, speed, and low cost of running experiments to assemble massive amounts of reproducable data.

To make the investigation more concrete, our game of choice is the Lemonade-stand Game (LSG), which consists of simple rules that illustrate an inherent tension between cooperation and competition [15]. This game is composed of three "lemonade vendors" who compete to serve the most customers in their vicinity, knowing that their competitors are looking to do the same. It is therefore a special case of a Hotelling game, which has been well studied in the economics literature [7, 10]. From a practical standpoint, location games like the LSG have obvious applications for retail establishments in a physical space, but they can also be applied in abstract spaces like on the web or social networks. Hotelling games have another interesting property, which is that they have a price of anarchy equal to $(2n-2)/n$ where $n$ is the number of players and the social optimum is considered to be the minimum score of the $n$ players [1]. Therefore, if many Nash equilibria exist, the resulting payoffs of an equilibrium can severely disadvantage any given player, which makes it all the more urgent for individuals to act according to an accurate population model in these settings.

In previous iterations of the tournament, the customers were spread evenly around the circular beach that functions as the environment and action set. This past year featured a new twist where the thirsty lemonade seekers are distributed unevenly, which creates a new payoff function and strategic challenge every time the game is played. One advantage of using this game as a testbed for studying new multiagent learning methods is that an annual tournament of submitted agents has been run in tandem with the Trading Agent Competition for the past two years. Because of this tournament, there is now a runnable library of agents that can be used for data collection, analysis, and model-building. The upcoming competition adds yet another twist, where the agents will be given the opportunity to learn over many matches (whereas before, the memory ended after one match of 100 rounds). Therefore, we would like a framework that prepares for this lifelong learning problem, while keeping data management to a minimum. This paper will present the CH model as one way to approach such a topic, which has led to an agent that won the 2011 LSG competition [15]. The results of this competition are given in Table 1.

The next section will discuss some of the most popular non-equilibrium focused behavioral models and gives details about the philosophy behind level-based reasoning. Section 3 presents a new framework for generating levels of reasoning in multiplayer games and learning the parameters of this model from observations. In Section 4, we present the Generalized Lemonade-stand Game, which has several properties that make it an intriguing test case for our extended model, and, using actual submitted agents from a LSG tournament, we perform experimental analysis in Section 5. Finally, we close with a discussion in Section 6.

| Place | Agent | Score | Std. Dev. |
|---|---|---|---|
| 1 | Rutgers (our agent) | 50.397 | ± 0.022 |
| 2 | Harvard | 48.995 | ± 0.020 |
| 3 | Alberta | 48.815 | ± 0.022 |
| 4 | Brown | 48.760 | ± 0.023 |
| 5 | Pujara | 47.883 | ± 0.020 |
| 6 | BMJoe | 47.242 | ± 0.021 |
| 7 | Chapman | 45.943 | ± 0.019 |
| 8 | GATech | 45.271 | ± 0.021 |

**Table 1: Official results of the 2011 Lemonade-stand Game Tournament. Our agent from Rutgers employs a version of the state-based Iterated Best Response (IBR) and won with a significant margin.**

## 2. BACKGROUND

Here, we present the models of behavior that inform our extended model. We should note that some of these models were developed primarily for two-player single-shot games in mind, and therefore need some adaptation for games with more players or that are played repeatedly.

### 2.1 Quantal Response

One proposed model takes into account the observation that decision makers choose actions according to some function of their value, as opposed to picking the best one. Under this procedure, agents' actions are seen to include some amount of error that can be quantified by the specified function. The most popular function goes by many names, including softmax, exponential weighting, Boltzmann exploration (in the reinforcement-learning literature) or logit quantal response (in game theory).

The quantal decision rule is as follows: A logit quantal action for an estimated utility function $u_i$ of agent $i$'s action $a_i$ results in mixed strategy $\pi_i$ given by

$$\pi(a_i) = \frac{e^{\lambda u_i(a_i)}}{\sum_{a'_i} e^{\lambda u_i(a'_i)}}$$

where $\lambda$ is defined as the exponential weighting parameter that decides how much error is expected depending on the relative action values.

In games, the utilities depend on opponent strategies, which are not specified by this model. The precision $\lambda$ is not specified either, and must be set or fit by the model designer. The quantal response mechanism does provide a convenient way to map values into actions that can be used to respond to opposing strategies.

### 2.2 Level-$k$

The basis for iterated reasoning models is the idea that agents perform various degrees of strategic reasoning. In this model, dubbed the level-$k$ model, strategies are formed in response to prior strategies known to implement some fixed reasoning capacity [5]. To be specific, an agent acting at level $k$ picks the best response to the strategy at the previous level. In the base case of level 0, the strategy is typically defined as a uniform random action, to provide the reasoning a base that does not perform any reasoning at all. This assumption can be justified by saying that if strategies resulting from this method cannot outperform random action selection, then they are probably not good in any real sense.

Given a set of actions $A$, Kronecker delta function $\delta : A \times A \rightarrow \{0,1\}$ and utility function $U : A_i \times A_{\neg i} \rightarrow \mathcal{R}$ mapping both agent $i$'s action and the strategies of rest of the population $\neg i$ to real values, the noiseless level-$k$ strategy $\pi_i^k$ of agent $i$ is

$$\begin{aligned}\pi_i^0(a_i) &= \frac{1}{|A|} \\ \pi_i^k(a_i) &= \delta(a_i, \arg\max_a u(a_i, \pi_{\neg i}^{k-1})).\end{aligned}$$

## 2.3 Quantal Level-$k$

This model combines the behaviors of the previous two models to arrive at a strategy calculation that incorporates the recursive best response of level-$k$ with the error robustness of quantal response. The quantal action selection operates on the values derived from the level-$k$ model at the desired setting of $k$:

$$\begin{aligned}\pi_i^0(a_i) &= \frac{1}{|A|} \\ \pi_i^k(a_i) &= \frac{e^{\lambda u(a_i, \pi_{\neg i}^{k-1})}}{\sum_{a_i'} e^{\lambda u(a_i', \pi_{\neg i}^{k-1})}}.\end{aligned}$$

## 2.4 Cognitive Hierarchy

We mention this alternative model to address a possible criticism of the level-$k$ model, which is that the best response step essentially ignores levels lower than $k-1$. This crucial point can lead to unwelcome phenomena such as repeating the mistakes of the past. For this reason the cognitive hierarchy model aims to respond to a distribution over previous strategies. Of course, model designers then face a new problem of how to identify the cumulative distribution over levels. The going standard for researchers is the Poisson distribution, which has the elegant property of derivation from a single parameter, $\tau$, that happens to coincide with the average level in the population [4].

If $P$ represents the Poisson function for some $\tau$, then let us define the Poisson-based Cognitive Hierarchy as

$$\begin{aligned}\pi_i^0(a_i) &= \frac{1}{|A|} \\ \pi_i^k(a_i) &= \delta\left(a_i, \arg\max_a \left(\sum_{\kappa=0}^{k-1} P(\kappa) u(a_{\neg i}, \pi_i^{\kappa-1}(a_{\neg i}))\right)\right).\end{aligned}$$

## 2.5 Algorithms for Modeling Agents

The recursive modeling framework has also had a big impact on the intersection between artificial intelligence and game theory. Recent formal models of reasoning paired with statistical inference include networks of multiagent influence diagrams [8] and interactive Partially Observable Markov Decision Processes [9]. These direct modeling algorithms work best and most efficiently against a single opponent, and can be used in repeated settings. Adding more players to the model adds a great deal of computational complexity, as the interactions proliferate and cris-crossing recursions appear. In contrast, the simpler behavioral economics models are well suited to groups of agents, where specific opponent modeling is not possible. The next section illuminates a process for adapting the idea of Iterated Best Response (IBR) for repeated games against more than a single agent.

| | Swerve | Straight |
|---|---|---|
| Swerve | 3, 3 | 1, 4 |
| Straight | 4, 1 | 0, 0 |

**Table 2: The game of Chicken.**

## 3. A STATE-BASED IBR FRAMEWORK

Consider the game of Chicken, for which the payoff bimatrix is given in Table 2. The payoffs of the game are such that a player is rewarded for playing the same action again and again because doing so encourages others to change their behavior to the fixed player's benefit (although the converse is also true). In such situations, the actions played by the participants take on the appearance of state, in which the players view the game both as a static environment in which reward must be maximized as well as a Chicken-like scenario where they might prefer to wait for others to back down. The initial positioning of the agents is often of supreme importance in these cases. We will use position or location and action interchangeably given the semi-fixed nature of the actions and for other reasons that will become clear as we develop the concrete example.

We next propose a model for repeated play that addresses these dual goals from the level-based perspective described previously. The decision-making process contains two phases: the initial action-selection problem, resembling a single-shot game with no history, and a state-based decision phase, which begins on the second round and continues until the end. The two phases resemble each other in significant ways, but differ in others. The first round is a special case of the state-based decision because there is no state information that exists; hence it is natural to use the typical assumption that others play randomly so that there is a way to initialize the strategic element. Because the initial action sets the stage for the remainder of the game, choosing a good start should not be ignored. The relationship between first and subsequent rounds is key to successfully identifying good opening moves.

## 3.1 Initial Action Levels

Although the approach we outline here parallels the one taken by previous game-theoretic models of behavior, we will diverge somewhat to handle cases with more than two agents. The level-$k$ model is successful in cases with only two players, call them Alex and Bill, because if Alex knows that Bill is playing level $k-1$, then of course it makes sense to respond with level $k$. However, what happens when Carla enters the game? Should the level-$k$ computation be optimized against two level $k-1$ players, or perhaps one level $k-1$ and one level 0 or something else entirely? We do not attempt to answer such questions here, but we would like to make a point about a certain class of games.

Consider a simple one-shot game with three players (Alex, Bill, and Carla) and three actions (1, 2, and 3) on a number line, so that 2 is connected to 1 and 3, but 1 is not connected to 3. The utilties of such a game for player $i$ are defined as

$$U(a_i) = I(a_i)/\#(a_i) + d(a_i, a_{\neg i})$$

where $I$ is the identity function, $\#(a_i)$ is the number of agents playing action $a_i$, and $d$ is a distance function that returns the minimum distance to the "closest" player in the action space identified by the numbers (i.e. $d(a_i, a_{\neg i}) = \min_{j|j \neq i}(|a_i - a_j|)$). If Carla were to calculate a strategy using level-based reasoning starting at random L0, she would find that L1 should play 3. At L2, the action choice depends on how Carla picks the likely population of Alex and Bill. If Carla believes there are two L1s playing action 3, then she has no preference over action 1 or 2 as they both get utility 3, as long as she does not pick 3 with a guaranteed score of 1. However, if she believes that Alex is an L1 but Bill is an L0, then she has a different choice. In this case, action 1 is worth $(0.5+2+3)/3 = 11/6$, action 2 is worth $(3+1+3)/3 = 7/3$, and action 3 is worth $(1.5 + 1.5 + 1)/3 = 4/3$. In either case, action 2 is optimal (for level 2). In the first case, the double L1 assumption caused Carla to completely misread the structure of the game, and ignore the crucial difference between action 1 and action 2, which is that action 2 has a higher base score.

Games like this one, such as certain location games, can mislead players into dismissing such differences if the lower strategies overlap too much. Therefore, caution dictates that diversity should trump purity when it comes to selecting the lower levels to optimize over. Sometimes it might work to use a distribution over these potential outcomes, but in other instances it may not be computationally possible to do so. In these cases, a mix of previous level and sub-previous level seems to balance out these concerns. Notice that this weakness is adequately addressed by modeling the population as a cognitive hierarchy distribution, but the problem of selecting a correct distribution leads to a multitude of possible responses. Using a cognitive hierarchy model raises computational difficulties, especially in repeated games.

## 3.2 State-based Levels

Once initial actions have been chosen and played, the position of others is known and the game takes on a state element. Now iterated reasoning separates into two paths. One is the opponent strategy given that our reasoner stays in place, and the other is the expected strategy once our reasoner changes its action. The resulting strategies will be useful in deciding whether or not a new action will exceed the existing cost of inaction. We make the distinction between the current action and a new action due to our realization that others are operating under the assumption that the existing state will likely persist in the future, and the reasoning process is thus continuing even though nothing has necessarily changed yet.

Another way of posing this problem is as the interaction between learning and teaching [11]. The two phenomena are linked because while it is beneficial for agents to learn a better course of action in regards to other agents, they must also understand how to influence the behavior of others.

It is in this context that the concept of regret will be useful to us. An algorithm's total regret is defined as the sum of differences of the total reward of the best strategy that can be adopted and the performance of the algorithm being evalutated up to the current point in time. It has been shown that certain adaptive algorithms have asymptotically no regret over a history of play [2], which suggests that no better strategy can be found. In our present model we would like to focus on a more limited definition of short-term regret as a way to balance the immediate realized sub-optimal past with a hoped-for improved future. Let $S$ denote the state, which is the set of actions chosen by each player. Let us define short-term regret of agent $i$ playing action $a_i$ in state $S$ as follows:

$$\mathcal{R}_{Si} = \max_{a \in A} \sum_{t=t_S}^{T} (u(a) - u(a_i)),$$

where $T$ is the current time and $t_S$ is the first timestep where the game was in the current state.

While an agent can easily minimize this regret by choosing its best response, it also needs to anticipate the reactions of its opponents, which is the purpose of the level-$k$ model. The current state provides a basis for the level computation if we assume that level 0 remains fixed with some unknown probability and otherwise acts randomly. In effect, this assumption leads to the maximizing best response to current state as level 1 as this response is the optimal move unless L0 happens to randomize. If we expect others to execute level 1 in a hypothetical changed state, this expected reaction allows a level 2 agent to compute the action that best prepares for that eventuality. At L2, agents act in a way that maximizes utility when the others eventually best respond. If this expectation is for some reason not met, it may result in inferior performance in the meantime.

A more sophistictated strategy type will aim to limit its vulnerability to this kind of outmaneuvering, in a way that is qualitatively different from simple best response. As such we will take the third level of reasoning to mean an equilibrium strategy. Putting a ceiling on the reasoning process is consistent with the earliest versions of recursive reasoning models which equate L3 with Nash behavior [12]. Reaching the final heights of the level-based action-selection does not necessarily complete a model for repeated games, however. In many games, there are no unique equilibrium strategies or outcomes, and players may wish to bring about better ones. In a wide class of games, this path takes the form of teaching other agents through a consistency of play in order to guide them towards desirable ends.

Suppose we are engaged in a repeated game of Chicken (see Table 2) with both players Driving Straight at each other. This situation is clearly sub-optimal for both: the regret-minimizing action, in a pure strategy sense, is for either player to Swerve. However, the reward-maximizing option is for only the opponent to Swerve so that we may enjoy the high reward. At any given moment, the choice comes down to this dilemma: do I try to wait out my opponent, in the hopes of getting the high score? Or do I succumb to the present urge to fold and take the smaller short-term gain? The only reason to wait in this scenario is due to the long-term summation of projected rewards contingent on the other player backing down. By discounting future rewards, the present value is finite regardless of how long the game continues. According to standard theory, the equation for current value $V$ of future fixed rewards $r_t = r$ received over an infinite time period and discounted by $\gamma$ per time period can be defined as

$$V = \frac{r}{1-\gamma}.$$

There comes a point where the regret overtakes this value for any given discount value $\gamma$. For a value-maximizing agent, this point marks the balance between maintaining a low utility position in hopes that another player will relent and relenting oneself. It is reasonable to expect that the current state of the game will persist for as long as it already has, due to the 50% principle, which states that the most likely temporal location for a random length occurence is halfway through the duration. The present accumulated regret serves to estimate the likely value lost during the upcoming time period, traded off against some optimistic future returns once that period ends. At that point the benefits of outwaiting the opponent are no longer as favorable as myopically choosing an action with higher utility than is currently received. From an observer's perspective, once an agent moves, the discount factor for that agent can be computed. One can also use previous observations to learn an ideal setting of the discount for optimal action.

In games like Prisoner's Dilemma, this waiting dilemma does not exist as there is no short-term gain for choosing to cooperate, nor is there a long-term cost for not doing so. Notice also that in this Chicken example the value $\gamma$ takes on a meta-game connotation. Denote the two players $W$ and $L$. Assume w.l.o.g. that $\gamma_W > \gamma_L$. Notice that $W$ gains a higher score, but the higher the value of $\gamma_L$, the worse both players perform. We can quantify this game by saying that, subtracting out the regret, the net value for the ultimate winner is $\frac{4}{1-\gamma_W} - \frac{4}{1-\gamma_L}$ whereas the loser gets $\frac{1}{1-\gamma_L} - \frac{4}{1-\gamma_L} = -\frac{3}{1-\gamma_L}$ at the moment the loser backs down.

Therefore, we propose the following method for discovering the ideal value of $\gamma$ given repeated observations of a population of agents in a game of this type. First, calculate $R_s$, the total regret experienced in the agent's previous state $s$, as the difference between expected experienced utility and potential utility in a new state $s'$. Next, calculate $\hat{U}_s$, the projected alternative score advantage received if it stays in state $s$ assuming the model prediction for the other agents holds. We can then set up this inequality to represent the tradeoff between known regret and potential gains at time $t = \tau$, which yields a condition on $\gamma$ where if this condition is broken, a different action should be taken.

$$\begin{aligned}
R_s &\geq \sum_{t=\tau}^{\infty} \gamma^t \hat{U}_s \\
\frac{R_s}{\hat{U}_s} &\geq \gamma^\tau + \sum_{t=\tau+1}^{\infty} \gamma^t \\
\frac{R_s}{\hat{U}_s} &\geq \gamma^\tau + \gamma \sum_{t=\tau+1}^{\infty} \gamma^{t+1} \\
\frac{R_s}{\hat{U}_s} &\geq \gamma^\tau + \gamma \sum_{u=\tau}^{\infty} \gamma^{u} \\
\frac{R_s}{\hat{U}_s} &\geq \gamma^\tau + \gamma \frac{R_s}{\hat{U}_s} \\
\frac{\gamma^\tau}{1-\gamma} &\leq \frac{\hat{U}_s}{R_s}
\end{aligned}$$

In practice, the discount factor can be interpreted as anything from model uncertainty to probability of events. These elements, once accurately learned, combine to form a model that can be effectively used to simulate behavior in previously unseen payoff settings over many rounds, prior to the first round of an interaction. This powerful predictive tool allows a user to identify the most advantageous position to stake out before others without this capability.

## 4. THE GENERALIZED LEMONADE-STAND GAME

The Lemonade Stand Game is a three-player game with simple rules, yet it gives rise to complex interaction patterns with cooperative as well as competitive elements. Imagine a sunny island with twelve beaches arranged like the numbers on a clock. Early in the morning three lemonade vendors set up their lemonade stand on one of the beaches in the dark (simultaneously) without knowing where the others will sell on that day. Their profit depends on the number of customers they attract, and each customer simply goes to the nearest lemonade stand (in case of a tie it goes to each nearest stand with equal probability).

Martin Zinkevich hosts a competition on an approximately annually basis, where he allows any participating team to submit a single agent [15]. These submissions make up the population of players in the competition. Each triplet plays the game and submissions are scored according to their average performance. The 2009 and 2010 competition featured a uniform customer distribution over the beach locations. In the most recent 2011 competition, agents competed in the Generalized Lemonade Stand Game, where each beach has one, two or three customers with equal probability. Each customer yields a profit of 6 for the stand that attracts it. In expectation, there is 144 cumulative payoff, and the lemonade stand positions decide how this is divided among the three lemonade vendors. This game is repeated for 100 days with the same customer distribution. An example draw with 150 cumulated payoff is given in Figure 1. Based on such a distribution and observations from the days that have passed, the three vendors need to choose their lemonade stand position for the next day. In each of these competitions, simple scripted strategies often outperform complex state-of-the-art online learning algorithms. Note that learning is only allowed within each game and agents are required to completely reset and purge all memory after every match.

### 4.1 Application of IBR

The Lemonade-Stand Game is a far richer environment than the Chicken or 3-player game described previously, but the same framework applies. This very game has been the

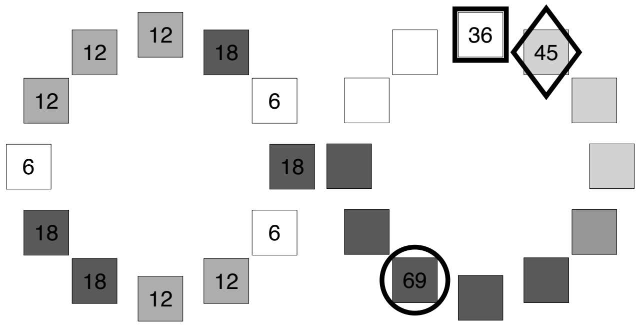


**Figure 1: The beaches on Lemonade Island are arranged like the numbers on the clock. Each beach has 1, 2 or 3 customers with equal probability, and each customer gives a profit of 6 (left, darker colors represent higher values). Given lemonade stands ◯, □ and ◊, the payoff to each can be computed (right, colors show to which stand customers go).**

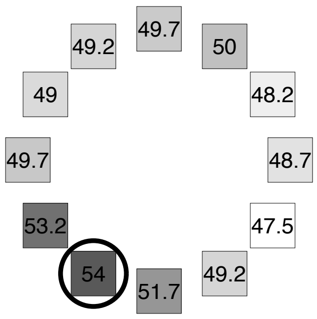


**Figure 2: Expected payoffs for reasoning Level 1: darker colors represent higher values. The circle denotes the best available action against two random players (L0).**

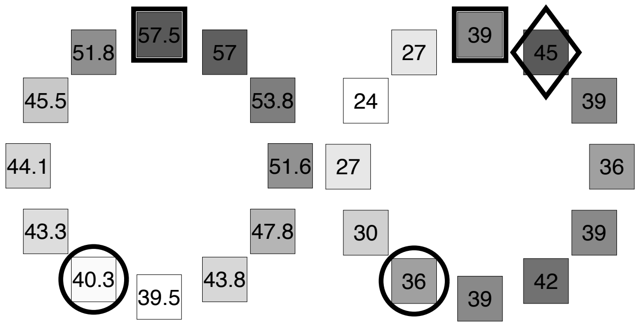


**Figure 3: Expected payoffs for reasoning Level 2 (left) and 3 (right): darker colors represent higher values. The □ denotes the best reply to one L1 player ◯ and one uniform random player (L0). The ◊ denotes the best available action against one L1 player ◯ and one L2 player □.**

subject of a few articles since the tournament began. Some participants have taken a more general approach to the planning elements [6] while others have addressed the way that iterative methods can produce the behaviors seen [14]. However, in the generalized version of the game, a richer model is required to handle the multitude of cases and viable potential strategies.

We assume an agent at Level 0 does not do any reasoning. Thus, it will chose an arbitrary action at random. An agent of the next reasoning level will assume its opponents both use reasoning Level 0. Such a player of Level 1 can compute its expected payoffs for all possible actions and choose the best reply (see Figure 2).

At all higher levels, the player will compute a best reply to opponents of the previous two levels. That is, a Level 2 player computes the expected payoffs against one Level 0 and one Level 1 opponent. Similarly, a Level 3 player computes the best reply to one Level 1 and one Level 2 opponent (see Figure 3 for an illustration). Structuring the iterative model in this way is a design choice, which is motivated both by the problems with overlapping strategies already mentioned and by the success in capturing the three-player interactions as presented in the following section.

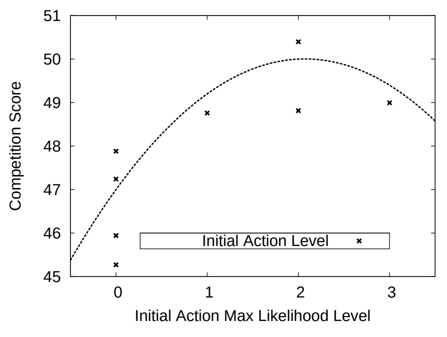
## 5. EXPERIMENTS AND VALIDATION

The experimental data presented here is derived from the submitted set of agents from the July 2011 Generalized LSG tournament, which implements the LSG with varying customer distributions. We have two goals in the following analysis. First, to show that the features resulting from the derived level-based model accurately predict performance in the tournaments. This is confirmed by the more advanced strategies under this definition being among the winners. Second, to demonstrate how this level data can be utilized to efficiently determine the distribution over types that are present in an ongoing series of games, and subsequently outflank them. This goal is of interest for upcoming competitions, where agents will have the chance to observe opponents in action over many games, albeit anonymously, and respond as they wish. As a side point, it has been observed that in LSG's current form, there is not much opportunity for modeling and responding to opponents within games, as matches are often settled within a small number of moves. Therefore any agent that possesses an accurate model of behaviors will have a great advantage when it comes to planning a strategy in matches with previously unseen density functions.

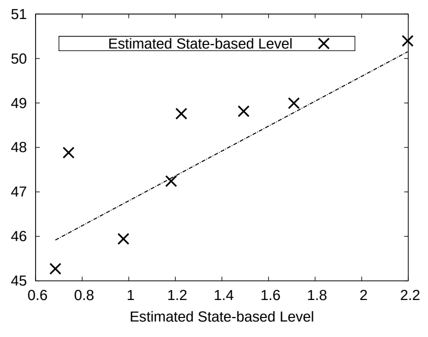
### 5.1 Learning Distributions over Levels

The quantal level-$k$ model (Section 2.3) provides a method for producing the strategies at each level given a precision value $\lambda$. To produce an estimate of the strategy executed by an agent, observe its action and the probability that each strategy would have selected it. Using this probability we arrive via Bayes at the likelihood that this agent is acting according to this strategy. The normalized likelihoods for all strategies give an estimated model of this agent. As described in the extended model 3, the two sets of observations that determine the distribution over levels are initial actions and new actions. Because the number of these observations may tend to be small in a single instance of the game at hand, we must gather data over many matches, with many different payoff functions. There are several ways to build models from this data. In the comprehensive survey and analysis done with human data [13], the authors used a maximum likelihood method to be able to make predictions of future behaviors. For our purposes, this predictive ability may not be enough because we would like to be able to generate strategies in response to a learned model. Therefore, a compromise solution is to simply average together the resulting likelihoods for every instance observed.

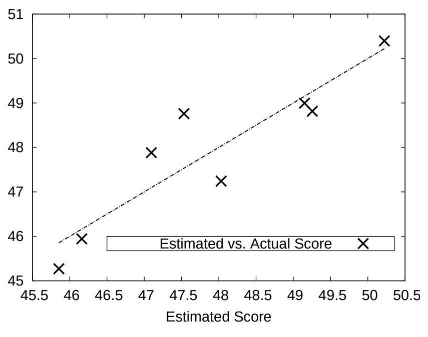
The mark of a successful model is to predict outcomes, and the state-based multiplayer IBR framework presented here is no exception. In this section we test the correlation between the various model parameters and final competition scores. Our results can be summarized in Figure 4. We took the level distributions for each agent and found the average estimated level for both initial action and state action in this manner. Then we examined the history leading up to a change in state and recorded the regrets of the participating agents, and thereby arrived at bounds on $\gamma$. This step gives three total model parameters, including the discount factor. All three are highly positively correlated with final performance. If we combine all three values to predict the score, we reach a 0.83 coefficient of correlation and over 0.9 for prediction of rankings. The initial action level has highest performance at L2, on account of other agents playing



(a)



(b)



(c)

**Figure 4: A comparison of the final competition score with the maximum likelihood level ($R^2 = 0.73$), estimated state-based level ($R^2 = 0.72$) and estimated score ($R^2 = 0.83$). (a) The best fit over the data points is a quadratic function because most of the agents are L1 or less, which means that L3 does worse than an L2 strategy in this population. (b) Average likelihoods of each level are computed and normalized, and then each level is weighted by the corresponding likelihood. The Spearman's rank coefficient is 0.93. (c) The estimated score is output by the final model as a function of initial level, state-based level, and discount factor. The Spearman's rank coefficient is 0.9. The trendline shown is X=Y.**

sub-L2 strategies. Recall that the level-$k$ model prescribes responding to the average degree of sophistication in the population, and not necessarily the highest possible strategy calculation with the most reasoning.

## 5.2 Building Strategies from a Learned Level-based Model

Once the strategy distribution is in place, we turn to finding optimal responses to the underlying population. The process used to accomplish this task is basically the mirror image of the learning mechanism. Given a new LSG instance, we discover the initial action levels, up to the third. In our case three is sufficient because the reasoning only needs to handle three agents. It so happens that by choosing one action from each of the three level strategies will yield a very good approximation to the likely final state of a series of best response steps, starting from any initial positions. In the event that the space cannot be divided in this way, there are probably a high number of equally good starting points distributed more or less evenly around the circle. We have done analysis to confirm this point but cannot show it for lack of room. We will suffice to call these actions *stable* for lack of a better word.

Once these three candidate actions are chosen, we can easily find the scores assuming that there is one agent at each location, giving a ranking of these three, which will not necessarily correspond to the level that first produced them. (See Figures 4-6 for a visual example explanation.) It is unlikely that the three players will end up picking a different one of the three stable actions. More likely is that at least one of our two opponents will choose the highest ranked action, as all else equal it would be the best one. Fortunately this situation is ideal for us to analyze using the regret-based approach mentioned in Section 3. If we have an estimated discount factor ($\gamma$) for our target population, then we know how long to expect an opponent to wait before switching to the lesser stable action, leaving our agent in command of the best one. If $\gamma$ is sufficiently low, it will be worth the initial cost to be the lucky agent to collect the higher score. However, if the population has demonstrated high $\gamma$ and therefore a lot of patience, then it may in fact be optimal to take one of the lower ranked stable actions, and hope that our opponents end up fighting over the highly ranked location. The parameterized model accounts for these opposing forces and combines them to compute the estimated values for each of these stable points over an entire game, prior to the game starting. Although we have described this process in words here, an agent we have built is able to quantitatively discover this solution automatically, and thus fully implement a model-based response.

In Table 3, we show the performance of our agent, the Full-Model agent, against the top four challengers in the latest tournament. The best submissions are used since other strategies have been observed to emulate them after some time [16], and we would like our agent to perform well against likely future competitors. This agent runs the model to estimate the game-length values of the best starting points, and selects the best of these accordingly. Once the game has begun, it switches into state-based mode and makes regret-informed decisions using an internal value of $\gamma$ that may be adjusted based on observations from the population. Table 4 replicates the original tournament with the addition of the Full-Model agent.

## 6. CONCLUSION

This article demonstrated a model synthesizing several

**Table 3: Results of an internal experimental match-up including the top four agents of the 2011 Tournament and an agent constructed using the full state-based IBR. 100 repetitions of each match shown.**

| Ranking | Agent | Score | Error |
|---|---|---|---|
| 1 | Full-Model | 51.61 | 0.039 |
| 2 | Rutgers | 49.36 | 0.037 |
| 3 | Alberta | 47.63 | 0.039 |
| 4 | Harvard | 47.60 | 0.032 |
| 5 | Brown | 43.10 | 0.034 |

**Table 4: Results of an internal experimental match-up including all agents of the 2011 Tournament and an agent constructed using the full state-based IBR. 100 repetitions of each match shown.**

| Ranking | Agent | Score | Error |
|---|---|---|---|
| 1 | Full-Model | 50.48 | 0.042 |
| 2 | Rutgers | 50.06 | 0.033 |
| 3 | Harvard | 49.21 | 0.035 |
| 4 | Alberta | 48.83 | 0.030 |
| 5 | Brown | 48.40 | 0.036 |
| 6 | Pujara | 47.27 | 0.039 |
| 7 | BMJoe | 46.95 | 0.037 |
| 8 | Chapman | 45.46 | 0.035 |
| 9 | GATech | 45.03 | 0.031 |

schools of thought regarding the modeling and prediction of agent behavior, especially including level-based reasoning and regret minimization. We built upon established methods for constructing a hierarchy of strategies through the tried-and-true best response operation as pieces of a robust process and adapted them to operate in new domains. Our present framework has a corresponding automated algorithm that outputs the strategies at each level. Given a data set of behavioral observations, the model infers the relevant parameters and frequencies of the constructed strategies. The final step is to translate the model into useable strategies, as we show through the Lemonade-stand Game example. The strength of our model is its compactness and ability to pre-simulate the likely course of action before the game is populated with agents, giving us a sizeable advantage when learning is allowed over many periods. The LSG tournament provides a case study for this model in action, and its past and present success is credited to the power of an automated procedure based on our framework.

**Algorithm 1** Model Pseudocode for LSG

```
Input: $\gamma$ (estimated optimal population discount factor)
Input: $P$ (estimated population level probabilities)
getAction()
  if $turn = 0$ then
    $stable[0] \leftarrow$ highest density location
    $stable[1] \leftarrow$ 2nd highest density location
    $stable[2] \leftarrow$ bestResponse($stable[0], stable[1]$)
    $max \leftarrow$ maxUtil($stable[0], stable[1], stable[2]$)
    $potRgrt \leftarrow$ utility of non-$max$ locations
    $collideRwrd \leftarrow$ utility if two players choose $max$
    $projRwrd \leftarrow (max - collideRwrd)$
    $futVal \leftarrow projRwrd/(1-\gamma)$
    $expRwrd \leftarrow$ comb. of $futVal$ and $projRwrd$ under $P$
    return $stable$ action with highest $expRwrd$
  else
    /* Use State-Based IBR */
    $cumRgrt \leftarrow cumRgrt +$ utilDiff(bestAlt, curConfig)
    $projRwrd \leftarrow$ utilDiff(curAction, newOppConfig)
    if $\frac{\gamma^{turn}}{1-\gamma} cumRgrt > projRwrd$ then
      $cumRgrt \leftarrow 0$
      return bestAlt in bestAltConfig
    else
      return curAction
    end if
  end if
```

## 7. REFERENCES

[1] A. Blum, M. T. Hajiaghayi, K. Ligett, and A. Roth. Regret minimization and the price of total anarchy. *In Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 373–382, 2008.

[2] M. Bowling. Convergence and no-regret in multiagent learning. *Advances in Neural Information Processing Systems 17 (NIPS)*, pages 209–216, 2005.

[3] C. F. Camerer. *Behavioral Game Theory: Experiments in Strategic Interaction.* Princeton University Press, 2003.

[4] C. F. Camerer, T.-H. Ho, and J.-K. Chong. A cognitive hierarchy model of games. *Quarterly Journal of Economics*, 119:861–898, 2004.

[5] M. Costa-Gomes, V. Crawford, and B. Broseta. Cognition and behavior in normal-form games: An experimental study. *Econometrica*, 69(5):1193–1235, 2001.

[6] E. M. de Côte, A. Chapman, A. M. Sykulski, and N. R. Jennings. Automated planning in adversarial repeated games. *UAI*, 2010.

[7] J. J. Gabszewicz and J.-F. Thisse. Location. *Handbook of Game Theory with Economic Applications*, 1992.

[8] Y. Gal and A. Pfeffer. Networks of influence diagrams: Reasoning about agents' beliefs and decision-making processes. *Journal of Artificial Intelligence Research (JAIR)*, 2008.

[9] P. Gmytrasiewicz and P. Doshi. A framework for sequential planning in multiagent settings. *Journal of AI Research (JAIR)*, 24:49–79, 2005.

[10] H. Hotelling. Stability in competition. *The Economic Journal*, 39:41Ű57, 1929.

[11] K. Leyton-Brown and Y. Shoham. *Multiagent Systems: Algorithmic, Game Theoretic and Logical Foundations.* Cambridge University Press, 2009.

[12] D. O. Stahl and P. W. Wilson. On players' models of other players: Theory and experimental evidence. *Games and Economic Behavior*, pages 218–254, 1995.

[13] J. R. Wright and K. Leyton-Brown. Beyond equilibrium: Predicting human behavior in normal form games. *The Twenty-Fourth Conference on Artificial Intelligence (AAAI-10)*, 2010.

[14] M. Wunder, M. Kaisers, M. Littman, and J. R. Yaros. Using iterated reasoning to predict opponent strategies. *International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2011.

[15] M. Zinkevich. The lemonade game competition. http://tech.groups.yahoo.com/group/lemonadegame/, December 2009.

[16] M. Zinkevich, M. Bowling, and M. Wunder. The lemonade stand game competition: Solving unsolvable games. *ACM SIGecom Exchanges*, 10, 2011.