Counter-Factual Reinforcement Learning: How To Model Decision-Makers That Anticipate The Future

Ritchie Lee, David H. Wolpert, James Bono, Scott Backhaus, Russell Bent, and Brendan Tracey

Abstract This paper introduces a novel framework for modeling interacting humans in a multi-stage game. This "iterated semi network-form game" framework has the following desirable characteristics: (1) Bounded rational players, (2) strategic players (i.e., players account for one another's reward functions when predicting one another's behavior), and (3) computational tractability even on real-world systems. We achieve these benefits by combining concepts from game theory and reinforcement learning. To be precise, we extend the bounded rational "level-K reasoning" model to apply to games over multiple stages. Our extension allows the decomposition of the overall modeling problem into a series of smaller ones, each of which can be solved by standard reinforcement learning algorithms. We call this hybrid approach "level-K reinforcement learning". We investigate these ideas in a cyber battle scenario over a smart power grid and discuss the relationship between

Ritchie Lee
Carnegie Mellon University Silicon Valley, NASA Ames Research Park, Mail Stop 23-11, Moffett Field, CA, 94035 e-mail: ritchie.lee@sv.cmu.edu

David H. Wolpert
Santa Fe Institute, 1399 Hyde Park Rd., Santa Fe, NM 87501
Los Alamos National Laboratory, MS B256, Los Alamos, NM, 87545
e-mail: david.h.wolpert@gmail.com

James Bono
American University, 4400 Massachusetts Ave. NW, Washington DC 20016
e-mail: bono@american.edu

Scott Backhaus
Los Alamos National Laboratory, MS K764, Los Alamos, NM 87545
e-mail: backhaus@lanl.gov

Russell Bent
Los Alamos National Laboratory, MS C933, Los Alamos, NM 87545 e-mail: rbent@lanl.gov

Brendan Tracey
Stanford University, 496 Lomita Mall, Stanford, CA 94305 e-mail: btracey@stanford.edu

the behavior predicted by our model and what one might expect of real human defenders and attackers.

1 Introduction

We are interested in modeling something that has never been modeled before: the interaction of human players in a very complicated time-extended domain, such as a cyber attack on a power grid, when the players have little or no previous experience with that domain. Our approach combines concepts from game theory and computer science in a novel way. In particular, we introduce the first time-extended level-K game theory model [9, 31, 37]. We solve this model using reinforcement learning (RL) algorithms [38] to learn each player's policy against the level $K-1$ policies of the other players. The result is a non-equilibrium model of a complex and time-extended scenario where bounded-rational players interact strategically. Our model is computationally tractable even in real-world domains.

1.1 Overview and Related Work

The foundation of our approach is the use of a "semi-Bayes net" to capture the functional structure of a strategic game. A semi-Bayes net is essentially a Bayes net [21] where the conditional probability distributions for nodes representing player decisions are left unspecified. Combining a semi-Bayes net with utility functions for the players yields a "semi network-form game" (or semi net-form game) [24], which takes the place of the extensive-form game [30] used in conventional game theory.[1] In this paper, we extend the semi net-form game framework to a repeated-time structure by defining an "iterated semi net-form game". The conditional probability distributions at the player decision nodes are specified by combining the iterated semi net-form game with a solution concept, e.g., the level-K RL policies used in this paper. The result is a Bayes net model of strategic behavior.

Like all Bayes nets, our model describes the conditional dependence relationships among a set of random variables. In the context of a strategic scenario, conditional dependencies can be interpreted to describe, for example, the information available to a player while making a strategic decision. In this way, semi net-form games incorporate a notion similar to that of "information sets" found in extensive-

[1] The "semi-" modifier refers to a restricted category of models within a broader class of models called network-form games. A key difference between the semi-network form game used here and the general formulation of network-form games is that the general formulation can handle unawareness – situation where a player does not know of the possibility of some aspect of the game [42]. Unawareness is a major stumbling block of conventional game theoretic approaches in part because it forces a disequilibrium by presenting an extreme violation of the common prior assumption [16].

form games. However, information in semi net-form games takes on the nature of information in statistics, thereby opening it to formal analysis by any number of statistical tools [22, 33] as opposed to information sets which uses an informal notion. Just as information sets are the key to capturing incomplete information in extensive-form games, conditional dependence relationships are the key to capturing incomplete information in semi net-form games.[2] In our example of a cyber battle, the cyber defender (power grid operator) has access to the full system state, whereas the cyber attacker only has access to the part of the system that has been compromised. Representing this in the semi net-form game diagram means the defender's decision node has the full system state as its parent, while the attacker's decision node only has a subset of the state as its parent. As a consequence, the attacker cannot distinguish between some of the system states. In the language of extensive-form games, we say that all states mapping to the same attacker's observation belong to the same information set.

It is important to recognize that the semi net-form game model is independent of a solution concept. Just as a researcher can apply a variety of equilibrium concepts (Nash equilibrium, subgame perfect equilibrium, quantal response equilibrium [27, 28], etc.) to the same extensive-form game, so too can various solution concepts apply to the same semi net-form game. In this paper we focus on the use of level-K RL policies, however, there is no way in which the semi net-form games model is dependent on that concept. One could, in principle, apply Nash equilibrium, subgame perfect equilibrium, quantal response equilibrium, etc. to a semi net-form game, though doing so may not result in a computationally tractable model or a good description of human behavior.

In the remainder of this introduction, we describe three characteristics whose unique combination is the contribution of our paper. The first is that players in our model are strategic; that their policy choices depend on the reward functions of the other players. This is in contrast to learning-in-games and co-evolution models [14, 20] wherein players do not use information about their opponents' reward function to predict their opponents' decisions and choose their own actions. On this point, we are following experimental studies [5], which routinely demonstrate the responsiveness of player behavior to changes in the rewards of other players.

Second, our approach is computationally feasible even on real-world problems. This is in contrast to equilibrium models such as subgame perfect equilibrium and quantal response equilibrium. We avoid the computational problems associated with solving for equilibria by using the level-K RL policy model, which is a non-equilibrium solution concept. That is, since level-K players are not forced to have correct beliefs about the actions of the other players, the level-K strategy of

[2] Harsanyi's Bayesian games [17] are a special case of extensive form games in which nature first chooses the game, and this move by nature generally belongs to different information sets for the different players. This structure converts the game of incomplete information to a game of imperfect information, i.e. the players have imperfectly observed nature's move. In addition to the fact that Harsanyi's used extensive form games in his work while we're using semi network-form games, our work also differs in what we are modeling. Harsanyi focused on incomplete information, while our model incorporates incomplete information and any other uncertainty or stochasticity in the strategic setting.

player i does not depend on the actual strategy of i's opponents. As a result, this means that the level-K RL policies of each of the players can be solved independently. The computational tractability of our model is also in contrast to partially observable Markov decision process- (POMDP-) based models (e.g. Interactive-POMDPs [15]) in which players are required to maintain belief states over belief states thus causing a quick explosion of the computational space. We circumvent this explosion of belief states by formulating policies as mappings from a player's memory to actions, where memory refers to some subset of a player's current and past observations, past actions, and statistics derived from those variables. This formulation puts our work more squarely in the literature of standard RL [18, 38]. As a final point of computational tractability, our approach uses the policy representation instead of the strategic representation of player decisions. The difference is that the policy representation forces player behavior to be stationary – the time index is not an argument of the policy – whereas in the strategic representation strategies are non-stationary in general.

Third, since our goal is to predict the behavior of real human players, we rely heavily on the experimental game theory literature to motivate our modeling choices. Using the policy mapping from memories to actions, it is straightforward to introduce experimentally motivated behavioral features such as noisy, sampled or bounded memory. The result of the RL, then, is an optimal strategy given more realistic assumptions about the limitations of human beings.[3] This is in contrast to the literature on coevolutionary RL [13, 29], where the goal is to find optimal strategies. For example, the work in [8] uses RL to design expert checkers strategies. In those models, behavioral features motivated by human experimental data are not included due to the constraining effect they have on optimal strategies. Hence, RL in our model is used as a description of how real humans behave. This use for RL has a foundation in neurological research [12, 25], where it has provided a useful framework for studying and predicting conditioning, habits, goal-directed actions, incentive salience, motivation and vigor [26]. The level-K model is itself another way in which we incorporate experimentally motivated themes. In particular, by using the level-K model instead of an equilibrium solution concept, we avoid the awkward assumption that players' predictions about each other are always correct [5, 19, 32].

We investigate all of this for modeling a cyber battle over a smart power grid. We discuss the relationship between the behavior predicted by our model and what one might expect of real human defenders and attackers.

[3] One can imagine an extension where the RL training is modified to reflect bounded rationality, satisfying [35], etc. For example, to capture satisficing, the RL may be stopped upon achieving the satisficing level of utility. Note that we do not pursue such bounded rational RL here.

1.2 Roadmap

This chapter is organized as follows. In Section 2, we provide a review of semi network-form games and the level-K d-relaxed strategies solution concept [24]. This review is the starting point for the theoretical advances of this paper found in Section 3. In Section 3 we extend the semi net-form games formalism to iterated semi network-form games, which enables interactions over a time-repeated structure. This is also where we introduce the level-K RL solution concept. Section 3 is the major theoretical contribution of this paper. In Section 4, we apply the iterated semi net-form game framework to model a cyber battle on a smart power distribution network. The goal of Section 4 is to illustrate how an iterated semi net-form game is realized and how the level-K RL policy solution concept is implemented. In this section we describe the setting of the scenario and lay out the iterated semi net-form game model, including observations, memories, moves and utility functions for both players. We also describe the details of the level-K RL algorithm we use to solve for players' policies. This section concludes with simulation results and a possible explanation for the resulting behaviors. Section 5 provides a concluding discussion of the iterated semi net-form games framework and future work.

2 Semi Network-Form Games Review

In this section, we provide a brief review of semi net-form games. For a rigorous treatment, please refer to Lee and Wolpert [24].

2.1 Framework Description

A "semi network-form game" (or semi net-form game) uses a Bayes net [21] to serve as the underlying probabilistic framework, consequently representing all parts of the system using random variables. Non-human components such as automation and physical systems are described using "chance" nodes, while human components are described using "decision" nodes. Formally, chance nodes differ from decision nodes in that their conditional probability distributions are prespecified. In contrast, each decision node is associated with a utility function which maps an instantiation of the net to a real number quantifying the player's utility. In addition to knowing the conditional distributions at the chance nodes, we must also determine the conditional distributions at the decision nodes to fully specify the Bayes net. We will discuss how to arrive at the players' conditional distributions over possible actions, also called their "strategies", later in Section 2.2. The discussion is in terms of countable spaces, but much of the discussion carries over to the uncountable case. We describe a semi net-form game as follows:

An (N**-player) semi network-form game** is described by a quintuple (G, X, u, R, π) where

1. G is a finite directed acyclic graph represented by a set of vertices and a set of edges. The graph G defines the topology of the Bayes network, thus specifying the random variables as well as the relationships between them.
2. X is a Cartesian product of the variable space of all vertices. Thus X contains all instantiations of the Bayes network.
3. u is a function that takes an instantiation of the Bayes network as input and outputs a vector in $\mathbb{R}^N$, where component i of the output vector represents player i's utility of the input instantiation. We will typically view it as a set of N utility functions where each one maps an instantiation of the network to a real number.
4. R is a partition of the vertices into $N+1$ subsets. The first N partitions contain exactly one vertex, and are used to associate assignments of decision nodes to players. In other words, each player controls a single decision node. The $N+1$ partition contains the remainder of the vertices, which are the chance nodes.
5. π is a function that assigns to every chance node a conditional probability distribution [21] of that node conditioned on the values of its parents.

Specifically, X_v is the set of all possible states at node v, u_i is the utility function of player i, $R(i)$ is the decision node set by player i, and π is the fixed set of distributions at chance nodes. Semi net-form game is a general framework that has broad modeling capabilities. As an example, a normal-form game [30] is a semi net-form game that has no edges. As another example, let v be a decision node of player i that has one parent, v'. Then the conditional distribution $P(X_v \mid X_{v'})$ is a generalization of an information set.

2.2 Solution Concept: Level-K D-Relaxed Strategies

In order to make meaningful predictions of the outcomes of the games, we must solve for the strategies of the players by converting the utility function at each decision node into a conditional probability distribution over that node. This is accomplished using a set of formal rules and assumptions applied to the players called a solution concept. A number of solution concepts have been proposed in the game theory literature. Many of which show promise in modeling real human behavior in game theory experiments, such as level-K thinking, quantal response equilibrium, and cognitive hierarchy. Although this work uses level-K exclusively, we are by no means wedded to this equilibrium concept. In fact, semi net-form games can be adapted to use other models, such as Nash equilibrium, quantal response equilibrium, quantal level-K, and cognitive hierarchy. Studies [5, 43] have found that performance of an equilibrium concept varies a fair amount depending on the game. Thus it may be wise to use different equilibrium concepts for different problems.

Level-K thinking [11] is a game theoretic solution concept used to predict the outcome of human-human interactions. A number of studies [2, 4, 5, 10, 11, 43] have

shown promising results predicting experimental data in games using this method. The concept of level-K is defined recursively as follows. A level K player plays (picks his action) as though all other players are playing at level $K-1$, who, in turn, play as though all other players are playing at level $K-2$, etc. This process continues until level 0 is reached, where the player plays according to a prespecified prior distribution. Notice that running this process for a player at $K \geq 2$ results in ricocheting between players. For example, if player A is a level 2 player, he plays as though player B is a level 1 player, who in turn plays as though player A is a level 0 player. Note that player B in this example may not be a level 1 player in reality – only that player A assumes him to be during his reasoning process. Since this ricocheting process between levels takes place entirely in the player's mind, no wall clock time is counted (we do not consider the time it takes for a human to run through his reasoning process). We do not claim that humans actually think in this manner, but rather that this process serves as a good model for predicting the outcome of interactions at the aggregate level. In most games, the player's level K is a fairly low number for humans; experimental studies [5] have found K to be somewhere between 1 and 2.

In [24], the authors propose a novel solution concept called "level-K d-relaxed strategies" that adapts the traditional level-K concept to semi network-form games. The algorithm proceeds as follows. To form the best response of a decision node v, the associated player $i = R^{-1}(v)$ will want to calculate quantities of the form $\text{argmax}_{x_v}[\mathbb{E}(u_i \mid x_v, x_{pa(v)})]$, where u_i is the player's utility, x_v is the variable set by the player (i.e., his move), and $x_{pa(v)}$ is the realization of his parents that he observes. We hypothesize that he (behaves as though he) approximates this calculation in several steps. First, he samples M candidate moves from a "satisficing" distribution (a prior distribution over his moves). Then, for each candidate move, he estimates the expected utility resulting from playing that move by sampling M' times the posterior probability distribution over the entire Bayes net given his parents and his actions (which accounts for what he knows and controls), and computing the sample expectation $\hat{u}_i^K$. Decision nodes of other players are assumed to be playing at a fixed conditional probability distribution computed at level $K-1$. Finally, the player picks the move that has the highest estimated expected utility. In other words, the player performs a finite-sample inference of his utility function using the information available to him, then picks (out of a subset of all his moves) the move that yields the highest expected utility. For better computational performance, the algorithm reuses certain sample sets by exploiting the d-separation property of Bayes nets [21]. The solution concept was used to model pilot behavior in a mid-air encounter scenario, and showed reasonable behavioral results.

3 Iterated Semi Network-Form Games

In the previous section, we described a method to model a single-shot scenario. That is, a scenario in which each player makes a single decision. However, most

real-world scenarios are not single-shot. Rather, what is typically seen is that the outcome is determined by a series of decisions made by each player over a time-repeated structure. One way to model time extension is to ignore the structure, create a large "rolled-out" net[4] that explicitly enumerates the repeated nodes, then apply level-K d-relaxed strategies described in Section 2.2. The problem with such an approach is that the roll-out causes a linear explosion in the number of decision nodes with the number of time steps. Since the computational complexity of level-K d-relaxed strategies is polynomial (to the Kth power) in the number of decision nodes [24], the algorithm becomes prohibitively slow in solving scenarios with more than a few time steps.

In this section, we extend the semi network-form game from Section 2 to an "iterated semi network-form game" (or iterated semi net-form game) in order to explicitly model the repeated-time structure of the game. Then we introduce a novel solution concept called "level-K reinforcement learning" that adapts level-K thinking to the iterated semi network-form game setting.

3.1 Construction of an Iterated Semi Network-Form Game

We describe the extended framework by building up the components incrementally. A "semi Bayes net" is like a standard Bayes net, in that a semi Bayes net has a topology specified by a set of vertices and directed edges, and variable spaces that define the possible values each vertex can take on. However, unlike a standard Bayes net, some nodes have conditional probability distributions (CPDs) specified, whereas some do not. The nodes that do not have their CPDs specified are decision nodes with one node assigned to each player. A pictorial example of a semi Bayes net is shown in Figure 1(a). The dependencies between variables are represented by directed edges. The oval nodes are chance nodes and have their CPDs specified; the rectangular nodes are decision nodes and have their CPDs unspecified. In this paper, the unspecified distributions will be set by the interacting players and are specified by the solution concept.

We create two types of semi Bayes nets: a "base semi Bayes net" and a "kernel semi Bayes net". A "base semi Bayes net" specifies the information available to all the players at the start of play, and is where the policy decisions of the game are made. Note that even though the game is time-extended, players only ever make one real decision. This decision concerns which policy to play, and it is made at the beginning of the game in the base semi Bayes net. After the policy decision is made, action decisions are merely the result of evaluating the policy at the current state. In contrast, the "kernel semi Bayes net" specifies both how information from the past proceeds to future instances of the players during play, and how the state of nature evolves during play. In particular, it specifies not only what a player currently

[4] Here we are violating the definition of a semi net-form game that each player can only control a single decision node. One way to deal with this is to treat past and future selves as different players, but having the same utility function.

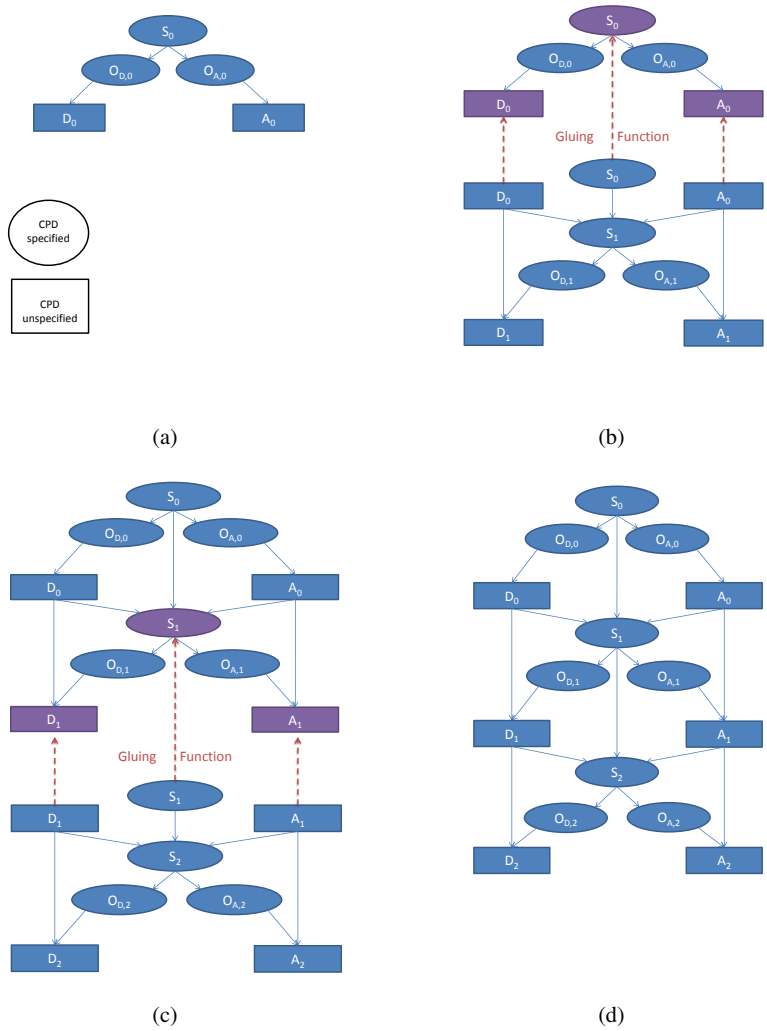

Fig. 1 Example construction of an iterated semi Bayes net with a base net and two kernels, i.e. $T = 2$, by repeatedly applying the "gluing" procedure. (a) A base semi Bayes net. (b) A kernel semi Bayes net being "glued" to a base semi Bayes net. (c) A second kernel semi Bayes net being appended to the net. (d) The final semi iterated Bayes net with $T = 2$. The numeric subscript indicates the time step to which each variable belongs.

observes, but also what they remember from their past observations and past actions. For example, the kernel semi Bayes net describes how the policy chosen in the base semi Bayes net is propagated to a player's future decision nodes, where a player's action choices are merely the result of evaluating that policy. From these two, we construct an "iterated semi Bayes net" by starting with the base semi Bayes net

then repeatedly appending the kernel semi Bayes net to it T times. Each append operation uses a "gluing" procedure that merges nodes from the first semi Bayes net to root nodes with the same spaces in the second semi Bayes net. Figure 1 illustrates how we build up an iterated semi Bayes net with a base net and two kernels, i.e. $T = 2$. Finally, we create an "iterated semi net-form game" by endowing an iterated semi Bayes net with a reward function, one for each player, defined at each time instant. The reward function takes as input an instantiation of the net at a particular (discrete) time and outputs a reward metric representing how happy the player is with that instantiation.[5]

3.2 Solution Concept: Level-K Reinforcement Learning

We introduce a novel solution concept for iterated semi net-form games that combines level-K thinking and reinforcement learning. Instead of considering all possible combinations of actions at individual decision nodes, we simplify the decision space by assuming that the players make only a single decision – what policy to play for the rest of the net. That is, the players pick a policy in the base semi Bayes net, and then executes that policy over all repetitions of the kernel semi Bayes net. This assumption allows us to convert the problem of computing a combination of actions over all time steps to one where we calculate a player's policy only once and reuse it T times. By reusing the policy, the computational complexity becomes independent of the total number of time steps. Formally, each unspecified node of a player contains two parts: A policy and an action. The policy is chosen in the base stage and is passed unchanged from the player's node in the base semi Bayes net to the player's node in the kernel semi Bayes net for all time steps. At each time step, the action component of the node is sampled from the policy based on the actual values of the node's parents. We point out that the utility of a particular policy depends on the policy decisions of other players because the reward functions of both players depend on the variables in the net.

The manner in which players make decisions given this coupling is specified by the solution concept. In this work we handle the interaction between players by extending standard level-K thinking from action space to policy space. That is, instead of choosing the best level K action (assuming other players are choosing the best level $K-1$ action), players choose the best level K policy (assuming that other players choose their best level $K-1$ policy). Instead of prespecifying a level 0 distribution over actions, we now specify a level 0 distribution over policies. Notice that from the perspective of a level K player, the behavior of the level $K-1$ opponents is identical to a chance node. Thus, to the player deciding his policy, the other players are just a part of his environment. Now what remains to be done is to calculate the best response policy of the player. In level-K reinforcement learning, we choose the

[5] We use the term reward function to conform to the language used in the RL literature. This is identical to the game theoretic notion of instantaneous utility (as opposed to the total utility, i.e. the present discounted value of instantaneous utilities).

utility of a player to be the sum of his rewards from each time step. In other words, the player selects the policy which leads to the highest expected infinite sum of discounted rewards. Noting this together with the fact that the actions of other players are identical to a stochastic environment, we see that the optimization is the same as a single-agent reinforcement learning problem where an agent must maximize his reward by observing his environment and choosing appropriate actions. There are many standard reinforcement learning techniques that can be used to solve such a problem [3, 18, 38]. The techniques we use in this paper are described in detail in Section 4.4.2.

For example, in a two-player iterated semi network-form game, the level 1 policy of player A is trained using reinforcement learning by assuming an environment that includes a player B playing a level 0 policy. If A is instead at level 2, his environment includes player B using a level 1 policy. Player A imagines this level 1 policy as having been reinforcement learned against a level 0 player A. To save computation time, it is assumed that how player B learns his level 1 distribution and how A imagines B to learn his level 1 distribution are identical.

4 Application: Cyber-Physical Security of a Power Network

4.1 Introduction

We test our iterated semi net-form game modeling concept on a simplified model of an electrical power grid controlled by a Supervisory Control and Data Acquisition (SCADA) system [39]. A SCADA system forms the cyber and communication components of many critical cyber physical infrastructures, e.g., electrical power grids, chemical and nuclear plants, transportation systems, and water systems. Human operators use SCADA systems to receive data from and send control signals to physical devices such as circuit breakers and power generators in the electrical grid. These signals cause physical changes in the infrastructure such as ramping electrical power generation levels to maintain grid stability or modifying the electrical grid's topology to maintain the grid's resilience to random component failures. If a SCADA system is compromised by a cyber attack, the human attacker may alter these control signals with the intention of degrading operations or causing permanent, widespread damage to the physical infrastructure.

The increasing connection of SCADA to other cyber systems and the use of computer systems for SCADA platforms is creating new vulnerabilities of SCADA to cyber attack [7]. These vulnerabilities increase the likelihood that the SCADA systems can and will be penetrated. However, even when a human attacker has gained some control over the physical components, the human operators still have some SCADA observation and control capability. The operators can use this capability to anticipate and counter the attacker moves to limit or deny the damage and maintain continuity of the infrastructure's operation. Traditional cyber security research on

cyber systems has focused on identifying vulnerabilities and how to mitigate those vulnerabilities. Here, instead, we assume that an attacker has penetrated the system, and we want to predict the outcome.

The SCADA attack and the defense by the SCADA operator can be modeled as a machine-mediated, human-human adversarial game. In the remainder of this section, we construct an iterated semi network-form game to model just such an interaction taking place over a simplified model of a SCADA-controlled electrical grid. The game is simulated using the level-K reinforcement learning solution concept described earlier. We explore how the strategic thinking embodied in level-K reinforcement learning affects the player performance and outcomes between players of different level K.

4.2 Scenario Model

Figure 2 shows a schematic of our simplified electrical grid infrastructure. It consists of a single, radial distribution circuit [40] starting at the low-voltage side of a transformer at a substation (node 1) and serving customers at nodes 2 and 3. Node 2 represents an aggregation of small consumer loads distributed along the circuit–such load aggregation is often done to reduce model complexity when simulating electrical distribution systems. Node 3 represents a relatively large, individually-modeled distributed generator located near the end of the circuit.

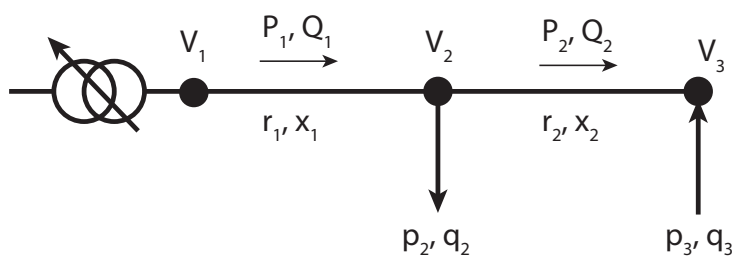

Fig. 2 Schematic drawing of the three-node distribution circuit consisting of three nodes i. The voltage at each node is V_i; the real and reactive power injections are p_i and q_i, respectively; the line reactance and resistance are x_i and r_i, respectively; and the real and reactive power flows in the distribution lines are P_i and Q_i, respectively.

In this figure, V_i, p_i, and q_i are the voltage and real and reactive power injections at node i. P_i, Q_i, r_i, and x_i are the real power flow, reactive power flow, resistance, and reactance of circuit segment i. These quasi-static power injections, power flows, voltages, and line properties are related by the nonlinear AC power flow equations [23]. Our focus in this work is on the game theoretic aspects of the model, therefore, we use a linearized description of the electrical power flow, i.e.,

the *LinDistFlow* equations [40]

$$P_2 = -p_3,\ \ Q_2 = -q_3,\ \ P_1 = P_2 + p_2,\ \ Q_1 = Q_2 + q_2 \tag{1}$$

$$V_2 = V_1 - (r_1 P_1 + x_1 Q_1),\ \ V_3 = V_2 - (r_2 P_2 + x_2 Q_2). \tag{2}$$

Here, all terms have been normalized by the nominal system voltage V_0 [23].

In this model, we assume that the circuit configuration is constant with $r_i = 0.03$ and $x_i = 0.03$. To emulate the normal fluctuations of consumer real load, p_2 is drawn from a uniform distribution over the range $[1.35, 1.5]$ at each time step of the game. The consumer reactive power is assumed to scale with real power, and we take $q_2 = 0.5p_2$ at each step of the game. The node 3 real power injection $p_3 = 1$ is also taken as constant implying that, although the distributed generator at node 3 is controllable (as opposed to a fluctuating renewable generator), its output has been fixed. Node 3 is then a reasonable model of an internal combustion engine/generator set burning diesel or perhaps methane derived from landfill gas. Such distributed generation is becoming more common in electrical distribution systems.

In our simplified game, the SCADA operator (defender) has one objective, i.e., keeping the voltages V_2 and V_3 within appropriate operating bounds (described in more detail below). To accomplish this the operator *normally* has two controls: 1) he can change the voltage V_1 at the head of the circuit, and 2) he can adjust the reactive power output q_3 of the distributed generator at node 3. However, we assume that the system has been compromised, and the attacker has taken control of q_3 while the defender retains control of V_1. In this circumstance, the *attacker* may use the injection of reactive power q_3 to modify all the Q_i causing the voltage V_2 to deviate significantly from 1.0. Excessive deviation of V_2 or V_3 can damage customer equipment [23] or perhaps initiate a cascading failure beyond the circuit in question. In the language of an iterated semi network-form game, the change in V_1 is the decision variable of the defender, q_3 is the decision variable of the attacker, and V_2, V_3, and the rest of the system state are determined by the *LinDistFlow* equations and probability distribution described above.

4.2.1 Players' Decision Spaces

In this scenario, the defender maintains control of V_1 which he can adjust in discrete steps via a variable-tap transformer [23], however, hardware-imposed limits constrain the defender's actions at time t to the following domain

$$D_{D,t} = \{\min(v_{max}, V_{1,t} + \delta v), V_{1,t}, \max(v_{min}, V_{1,t} - \delta v)\} \tag{3}$$

where δv is the voltage step size for the transformer, and v_{min} and v_{max} represent the absolute min and max voltage the transformer can produce. In simple terms, the defender may leave V_1 unchanged or move it up or down by δv as long as V_1 stays within the range $[v_{min}, v_{max}]$. In our model, we take $v_{min} = 0.90$, $v_{max} = 1.10$, and $\delta v = 0.02$. Similarly, hardware limitations of the generator at node 3 constrain

the attacker's range of control of q_3. In reality, the maximum and minimum values of q_3 can be a complicated function [23] of the maximum real power generation capability $p_{3,max}$ and the actual generation level p_3. To keep the focus on the game theoretic aspects of the model, we simplify this dependence by taking the attacker's q_3 control domain to be

$$D_{A,t} = \{-q_{3,max}, \ldots, 0, \ldots, q_{3,max}\}, \tag{4}$$

with $q_{3,max} = p_{3,max}$. To reduce the complexity of the reinforcement learning computations, we also discretize the attacker's move space to eleven equally-spaced settings with $-q_{3,max}$ and $+q_{3,max}$ as the end points. Later, we study how the behavior and performance of the attacker depends on the size of the assets under his control by varying p_3 from 0 to 1.8.

4.2.2 Players' Observed Spaces

The defender and attacker make observations of the system state via the SCADA system and the attacker's compromise of node 3, respectively. Via the SCADA system, the defender retains wide system visibility of the variables important to his operation of the system, i.e., the defender's observed space is given by

$$\Omega_D = [V_1, V_2, V_3, P_1, Q_1, \mathcal{M}_D]. \tag{5}$$

Because he does not have access to the full SCADA system, the attacker's observed space is somewhat more limited

$$\Omega_A = [V_2, V_3, p_3, q_3, \mathcal{M}_A]. \tag{6}$$

Here, $\mathcal{M}_D$ and $\mathcal{M}_A$ each denote real numbers that represent a handcrafted summary metric of the respective player's memory of the past events in the game. These are described in Section 4.2.4.

4.2.3 Players' Rewards

The defender desires to maintain a high quality of service by controlling the voltages V_2 and V_3 near the desired normalized voltage of 1.0. In contrast, the attacker wishes to damage equipment at node 2 by forcing V_2 beyond normal operating limits. Both the defender and attacker manipulate their controls in an attempt to maximize their own average reward, expressed through the following reward functions

$$R_D = -\left(\frac{V_2 - 1}{\epsilon}\right)^2 - \left(\frac{V_3 - 1}{\epsilon}\right)^2, \tag{7}$$

$$R_A = \Theta(V_2 - (1+\epsilon)) + \Theta((1-\epsilon) - V_2). \tag{8}$$

Here, ϵ represents the halfwidth of the nominally good range of normalized voltage. For most distribution systems under consideration, $\epsilon \sim 0.05$. $\Theta(\cdot)$ is the step function.

4.2.4 Players' Memory Summary Metrics

The defender and attacker use memory of the system evolution in an attempt to estimate part of the state that is not directly observable. In principle, player memories should be constructed based on specific application domain knowledge or interviews with such experts. However, in this initial work, we simply propose a memory summary metric for each player that potentially provides him with additional, yet imperfect, system information. We define the defender memory summary metric to be

$$\mathcal{M}_{D,t} = \frac{1}{m+1} \sum_{n=t-m}^{t} \text{sign}(V_{1,n} - V_{1,n-1}) \, \text{sign}(V_{3,n} - V_{3,n-1}) \tag{9}$$

If the attacker has very limited q_3 capability, both p_3 and q_3 are relatively constant, and *changes* in V_3 should follow *changes* in V_1, which is directly controlled by the defender. If all V_3 changes are as expected, then $\mathcal{M}_D = 1$. The correlation between V_1 and V_3 changes can be broken by an attacker with high q_3 capability because large changes in q_3 can make V_1 and V_3 move in opposite directions. If attacker actions always cause V_1 and V_3 to move in opposite directions, then $\mathcal{M}_D = -1$. This correlation can also be broken by variability in the (unobserved) p_2 and q_2. The attacker could use this (p_2, q_2) variability, which is unobserved by the attacker, to mask his actions at node 3. Such masking is more important in a setting where the defender is uncertain of the presence of the attacker, which we will address in future work.

As with the defender memory summary metric, the intent of $\mathcal{M}_A$ is to estimate some unobserved part of the state. Perhaps the most important unobserved state variable for the attacker is V_1 which reveals the vulnerability of the defender and would be extremely valuable information for the attacker. If the attacker knows the rules that the defender must follow, i.e., Equation (3), he can use his observations to infer V_1. One mathematical construct that provides this inference is

$$\mathcal{M}_{A,t} = \sum_{n=t-m}^{t} \text{sign}\left(\text{floor}\left(\frac{\Delta V_{3,n} - \Delta q_{3,n} x_2 / V_0}{\delta v}\right)\right). \tag{10}$$

If the attacker increases q_3 by $\Delta q_{3,t} = q_{3,t} - q_{3,t-1}$, he would expect a proportional increase in V_3 by $\Delta V_{3,t} = V_{3,t} - V_{3,t-1} \sim \Delta q_3 x_2 / V_0$. If V_3 changes according to this reasoning, then the argument in $\mathcal{M}_A$ is zero. However, if the defender adjusts V_1 at the same time step, the change in V_3 would be modified. If $\Delta V_{3,t}$ is greater or lower than the value expected by the attacker by $\Delta V/N$, the argument in $\mathcal{M}_A$ is +1 or -1, respectively. The sum then keeps track of the net change in V_1 over the previous m time steps. Note also that the stochastic load (p_2, q_2) will also cause changes in V_3 and, if large enough, it can effectively mask the defender behavior from the attacker.

4.3 Iterated Semi Network-Form Game Model

We model the scenario described in Section 4.2 as an iterated semi net-form game set in the graph shown in Figure 3. The figure shows the net for 2 time steps with the numeric subscript on each variable denoting the time step to which it belongs. The system state $S = [P_2, Q_2, P_1, V_1, V_2, V_3]$ is a vector that represents the current state of the power grid network. The vector comprises of key system variables with their relationships defined in Equations (1) and (2). The observation nodes $O_D = [V_1, V_2, V_3, P_1, Q_1]$ and $O_A = [V_2, V_3, p_3, q_3]$ are vectors representing the part of the system state that is observed by the defender and attacker, respectively. We compute these observation nodes by taking the system state S, and passing through unchanged only the variables that the player observes. Each player's observation is incorporated into a memory node (M_D and M_A for the defender and attacker, respectively) that summarizes information from the player's past and present. The memory nodes[6] are given by $M_{D,t} = [O_D, \mathcal{M}_{D,t}, D_{D,t-1}]$ and $M_{A,t} = [O_{A,t}, \mathcal{M}_{A,t}, D_{A,t-1}]$. Now, the defender uses his memory M_D to set the decision node D_D, which adjusts the setting of the voltage-tap transformer (up to one increment in either direction) and sets the voltage V_1. On the other hand, the attacker uses his memory M_A to set the decision node D_A, which sets q_3. Finally, the decisions of the players are propagated to the following time step to evolve the system state. In our experiments we repeat this process for $T = 100$ time steps.

4.4 Computing the Solution Concept

We compute the level-K policies of the players following the level-K reinforcement learning solution concept described in Section 3.2. First, we form the base of the level-K hierarchy by defining level 0 policies for the defender and attacker. Then, we describe the details of how we apply reinforcement learning to bootstrap up to levels $K > 0$. A level 0 policy represents a prior on the player's policy, i.e., it defines how a non-strategic player would play. In this work, we handcrafted level 0 policies based on expert knowledge of the domain. In future work, we would like to devise an automated and somewhat "principled" way of setting the level 0 policies.

4.4.1 Level 0 Policies

Often, level 0 players are assumed to choose their moves randomly from their move spaces $D_{D,t}$ and $D_{A,t}$. However, we do not believe this to be a good assumption, especially for SCADA operators. These operators have training which influences how

[6] To be technically correct, we must also include the variables carried by the memory nodes M_D and M_A for the sole purpose of calculating $\mathcal{M}_D$ and $\mathcal{M}_A$, respectively. However, for simplicity, we are not showing these variables explicitly.

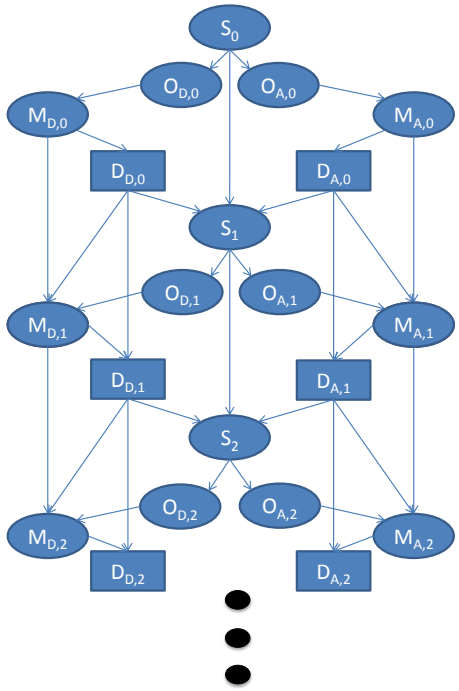

Fig. 3 The iterated semi net-form game graph of the cyber security of a smart power network scenario. The graph shows 2 time steps explicitly. In our experiments we choose the number of time steps $T = 100$. We use subscripts D and A to denote node association with the defender and attacker, respectively, and the numeric subscript to denote the time step. The system state S represents the current state of the power grid network. The players make partial observations O of the system and use them to update their memories M. The memories are used to pick their action D.

they control the system when no attacker is present, i.e., the "normal" state. In contrast, a random-move assumption may be a reasonable model for a level 0 attacker that has more knowledge of cyber intrusion than of manipulation of the electrical grid. However, we assume that our level 0 attacker also has some knowledge of the electrical grid.

If there is no attacker present on the SCADA system, the defender can maximize his reward by adjusting V_1 to move the average of V_2 and V_3 closer to 1.0 without any concern for what may happen in the future. We take this myopic behavior as representative of the level 0 defender, i.e.,

$$\pi_D(V_{2,t}, V_{3,t}) = \arg\min_{D_{D,t}} \frac{(V_{2,t} + V_{3,t})}{2} - 1 \tag{11}$$

For the level 0 attacker, we adopt a *drift-and-strike* policy which requires some knowledge of the physical circuit and power flow equations. We propose that the attacker "drifts" in one direction by steadily increasing (or decreasing) q_3 by one increment at each time step. The level 0 attacker decides the direction of the drift based on V_2, i.e., the attacker drifts to larger q_3 if $V_2 < 1$. The choice of V_2 to decide the direction of the drift is somewhat arbitary. However, this is simply assumed level 0 attacker behavior. The drift in q_3 causes a drift in Q_1 and, without any compensating move by the defender, a drift in V_2. However, a level 0 defender compensates by drifting V_1 in the opposite sense as V_2 in order to keep the average of V_2 and V_3 close to 1.0. The level 0 attacker continues this slow drift until, based on his knowledge of the power flow equations and the physical circuit, he detects that a sudden large change in q_3 in the opposite direction of the drift would push V_2 outside the range $[1-\varepsilon, 1+\epsilon]$. If the deviation of V_2 is large enough, it will take the defender a number of time steps to bring V_2 back in range, and the attacker accumulates reward during this recovery time. More formally this level 0 attacker policy can be expressed as

Level0Attacker()

$V^* = \max_{q \in D_{A,t}} |V_2 - 1|$;
if $V^* > \theta_A$
  then return $\arg\max_{q \in D_{A,t}} |V_2 - 1|$;
if $V_2 < 1$
  then return $q_{3,t-1} + 1$;
return $q_{3,t-1} - 1$;

Here, θ_A is the threshold parameter that triggers the strike. Throughout this work, we have used $\theta_A = 0.07 > \epsilon$ to indicate when an attacker strike will accumulate reward.

4.4.2 Reinforcement Learning Details

The training environment of a level-K player consists of all nodes that he does not control, including all chance nodes and the decision nodes of other players, which are assumed to be playing with a level $K-1$ policy. This leaves us with a standard single-agent reinforcement learning problem, where given an observation, the player must choose an action to maximize some notion of cumulative reward. We follow loosely the SARSA reinforcement learning setup in [38]. First, we choose the optimization objective to be his expected sum of discounted single-step rewards (given by Equations 7 and 8). To reduce the output space of the player, we impose an ε-greedy parameterization on the player's policy space. That is, the player plays what he thinks is the "best" action with probability $1-\varepsilon$, and plays uniformly randomly over all his actions with probability ε. Playing all possible actions with nonzero probability ensures sufficient exploration of the environment space for learning. At the core of the SARSA algorithm is to learn the "Q-function", which is a mapping from observations and actions to expected sum of discounted rewards (also known as "Q-values"). Given an observation of the system, the Q-function gives the

long-term reward for playing a certain action. To maximize the reward gathered, the player simply plays the action with the highest Q-value at each step.

To learn the Q-function, we apply the one-step SARSA on-policy algorithm in [38].[7] However, since the players' input spaces are continuous variables, we cannot use a table to store the learned Q-values. For this reason, we approximate the Q-function using a neural-network [3, 34]. Neural networks are a common choice because of its advantages as a universal function approximator and being a compact representation of the policy.

To improve stability and performance, we make the following popular modifications to the algorithm: First, we run the algorithm in semi-batch mode, where training updates are gathered and updated at the end of the episode rather than following each time step. Second, we promote initial exploration using optimistic starts (high initial Q-values) and by scheduling the exploration parameter ε to a high rate of exploration at first, then slowly decreasing it as the training progresses.

4.5 Results and Discussion

Level-K reinforcement learning was performed for all sequential combinations of attacker and defender pairings, i.e., D1/A0, D2/A1, A1/D0, and A2/D1. Here, we refer to a level K player using a shorthand where the letter indicates attacker or defender and the number indicates the player's level. The pairing of two players is indicated by a "/". The training was performed for $q_{3,max}$ in the range 0.2 to 1.8. Subsequent to training, simulations were run to assess the performance of the different player levels. The player's average reward per step for the different pairs is shown in Figure 4 as a function of $q_{3,max}$. Figure 5 shows snapshots of the players' behavior for the pairings D0/A0, D1/A0, and D0/A1 for $q_{3,max}$ =0.7, 1.2, and 1.6. Figure 6 shows the same results but for one level higher, i.e., D1/A1, D2/A1, and D1/A2.

D0/A0

Figures 5(b), (e), and (h) show the interaction between the two level 0 policies, and Figures 4(a) and (d) show the average player performance. These initial simulations set the stage for interpreting the subsequent reinforcement learning. For $q_{3,max} < 0.8$, the black circles in Figure 4(d) show that A0 is unable to push V_2 outside of the range $[1-\epsilon, 1+\epsilon]$. The explanation is found in Figure 5(b). With $V_2 < 1$ and say $q_{3,max} = 0.7$, A0's drift will have saturated at $q_3 = q_{3,max} = 0.7$. However, with $\theta_{\mathcal{A}} = 0.07$, A0 will not strike by changing $q_3 = -q_{3,max} = -0.7$ unless he projects such a strike could drive V_2 below 0.93. A0's limited q_3-strike capability is not enough

[7] Singh et al. [36] describes the characteristics of SARSA when used in partially observable situations. SARSA will converge to a reasonable policy as long as the observed variables are reasonably Markov.

overcome the threshold and the system becomes locked in a quasi-steady state. In the midrange of A0's capability ($0.8 \leq q_3 \leq 1.4$), the drift-and-strike A0 policy is effective (Figure 5(e)). However, A0 is only successful for strikes that force $V_2 < 0.95$. In addition, there are periods of time when $V_2 \sim 1.0$ and A0 is unable to decide on a drift direction. However, these become fewer (and A0's average reward grows) as $q_{3,max}$ approaches 1.4 (Figure 4(d)). For $q_{3,max} \geq 1.6$, A0 is able to successfully strike for $V_2 < 0.93$ and $V_2 > 1.07$, and A0 drives the system into a nearly periodic oscillation (Figure 5(h)) with a correspondingly large increase in A0's discounted average reward (Figure 4(d)). The reduction in D0's performance closely mirrors the increase in A0's performance as q_3 increases. However, *it is important to note that D0 enables much of A0's success by changing V_1 to chase the V_2 and V_3*. The adjustments in V_1 made by D0 in Figures 5(b), (e), and (h) bring the system closer to the voltage limits just as A0 gains a large strike capability.

D1 Training Versus A0

The red triangles in Figure 4(a) and the black circles in Figure 4(e) show dramatic improvement in the performance of D1 over D0 when faced with A0. In the middle range of A0's capability ($0.8 \leq q_{3,max} \leq 1.4$), Figure 5(d) shows that D1 stops changing V_1 to chase the immediate reward sought by D0. Instead, D1 maintains a constant $V_1 = 1.02$ keeping $V_2 \sim 1.0$ and A0 uncertain about which direction to drift. By keeping $V_1 > 1.0$, D1 also corrects the error of D0 whose lower values of V_1 helped A0 push V_2 and V_3 below $1 - \epsilon$. With $V_1 = 1.02$, the average of V_2 and V_3 are significantly higher than 1.0, but D1 accepts the immediate decrement in average reward to avoid a much bigger decrement he would suffer from an A0 strike. The effect of this new strategy is also reflected in the poor A0 performance as seen from the black circles in Figure 4(e). The behavior of D1 for $q_{3,max} \geq 1.6$ in Figure 5(g) becomes complex. However, it appears that D1 has again limited the amount that he chases V_2 and V_3. In fact, D1 moves V_1 in a way that decreases his immediate reward, but this strategy appears to anticipate A0's moves and effectively cuts off and reverses A0 in the middle of his drift sequence. We note that this behavior of the defender makes sense because he knows that the attacker is there waiting to strike. In real life, a grid operator may not realize that a cyber attack is even taking place. To capture this phenomenon motivates follow-on work in uncertainty modeling of the attacker's existence.

A1 Training Versus D0

A cursory inspection of Figures 5(c), (f), and (i) might lead one to believe that the A1 training has resulted in A1 simply oscillating q_3 back and forth from $+q_{3,max}$ to $-q_{3,max}$. However, the training has resulted in rather subtle behavior, which is most easily seen in Figure 5(c). The largest change A1 (with $q_{3,max} = 0.7$) can independently make in V_2 is ~ 0.04. However, A1 gains an extra 0.02 of voltage change by

leveraging (or perhaps convincing) D1 to create oscillations of V_1 in-phase with his own moves. For this strategy to be effective in pushing V_2 below $1-\epsilon$, the V_1 oscillations have to take place between 1.0 and 1.02, or lower. When the synchronization of the V_1 and A1 oscillations are disturbed such as at around step 50 in Figure 5(c), A1 modifies his move in the short term to delay the move by D0 and re-establish the synchronization. A1 also appears to have a strategy for "correcting" D0's behavior if the oscillations take place between levels V_1 that are too high. Near step 40 in Figure 5, A1 once again delays his move convincing D0 to make two consecutive downward moves of V_1 to re-establish the "correct" D0 oscillation level. Similar behavior is observed out to $q_{3,max} = 1.4$. At $q_{3,max} = 1.6$, A1 has enough capability that he can leverage in-phase D0 oscillations to exceed both the V_2 lower and upper voltage limits. This improved performance is reflected in the dramatic increase in A1's average reward (A1/D0; see red triangles in Figure 4(d)).

D1/A1

In the hierarchy of level-K reinforcement learning, D1/A1 is similar to D0/A0 in that they do not train against one another, but this match up sets the stage for interpreting the level-2 trainings. Figures 5(a), (d), and (g) show that the D1/A0 training results in a D1 that does not chase V_2 and V_3, keeps V_2 near 1.0, and accepts a lower current reward to avoid large A0 strikes. In Figures 6(b), (e), and (h), D1 continues to avoid responding to the oscillatory behavior of A1, V_2 generally does not cross beyond the acceptable voltage limits. However, V_3 is allowed to deviate significantly beyond the bounds. The result is that D1's average reward versus A1 does not show much if any improvement over D0's versus A1 (red triangles and black circles, respectively, in Figure 4(b)). However, D1 is quite effective and reducing the performance of A1 (Figures 4(e) red triangles) relative to the performance of A1 in D0/A1, at least for the intermediate values of $q_{3,max}$ (Figure 4(d) red triangles). The results for A1 are clearer. Figures 6(b), (e), and (h) show the oscillatory behavior of A1 while Figures 4(a), (b), (d), and (e) show that the switch from A0 to A1 when facing D1 improves the attacker's performance while degrading the performance of D1.

D2 Training Versus A1

The results of this training start out similar to the training for D1. Figure 6(a) shows that, at $q_{3,max} = 0.7$, D2 performs better if he does not make many changes of V_1 thereby denying A1 the opportunity to leverage his moves to amplify the swings of V_2. For the higher values of $q_{3,max}$ in Figures 6(d) and (g), D2 learns to anticipate the move pattern of A1 and moves in an oscillatory fashion, but one that is *out of phase* with the moves of A1. Instead of amplifying the swings of V_2, D2's moves attenuate these swings. This new behavior results in across-the-board improvement in D2's average discounted reward over D1 (blue squares versus red triangles in Figure 4(b)

and a significant reduction in A1 performance (red triangles in Figure 4(e) versus Figure 4(f)).

A2 Training Versus D1

A2 shows no perceptible increase in performance over A1 when matched against D1 (blue squares versus red triangles in Figure 4(e)). The same general observation can be made for A2 and A1 when matched against any of D0, D1, or D2. Figures. 4(b) and (c) show that the defenders perform nearly the same against A1 or A2, and Figures 4(e) and (f) show no significant change in attacker performance when switching from A1 to A2. This may indicate that policies embodied in A2 (or A1) may be approaching a fixed point in performance.

D2/A2

The similarities in the performance of A1 and A2 make the analysis of this interaction nearly the same as that of D2/A1.

5 Conclusions and Future Work

In this paper, we introduced a strategic, computationally-tractable, experimentally-motivated model for predicting human behavior in novel and complex time-extended scenarios. This model consists of an iterated semi net-form game combined with a level-K RL solution concept. We applied this model to predict behavior on a cyber battle on a smart power grid. As discussed in the results section, the predictions of this model are promising in that they match expectations for how a "real world" cyber battle would unfold.

We can vary parameters of the model that both concern the kind of cyber battle taking place (e.g., degree of compromise) and that describe the players (e.g., level 0 distributions, their level K). We can also vary the control algorithm. We can then evaluate the expected "social welfare" (i.e., the happiness metric of the system designer) for all such variations. In this way our framework can be used to increase our understanding of existing and proposed control algorithms to evaluate their robustness under different cyber attack scenarios and/or model mis-specification. In the near future, with additional advances in our computational algorithms, we hope to be able to solve the model in real-time as well. This raises the possibility of using our framework to do real-time control rather than choose among some small set of proposed control algorithms, i.e., to dynamically predict the attacker's policy and respond optimally as the cyber battle unfolds.

Despite the significant modeling advances presented here, there are several important ways in which the realism of this paper's model can be improved. Some of

these improvements have already been formalized, but they were left out of this document for the purposes of space and clarity. For example, the iterated semi net-form game framework easily models the situation where players have uncertainty about the environment they are facing. This includes uncertainty about the utility functions and the rationality (or levels) of the other players. This naturally corresponds to the Bayesian games setting within the extensive form games formalism. This also includes uncertainty about whether or not the other players exist. In fact, the semi net-form game formalism is unique in that it can even be extended to handle "unawareness" – a situation where a player does not know of the possibility of some aspect of the game. For example, it would be unawareness, rather than uncertainty, if the defender did not know of the possibility that an attacker could take control of a portion of the smart power grid. These types of uncertainty and unawareness will be presented and explored in future work.

Another important modeling advance under development is related to the ability of players to adapt their policies as they interact with their opponents and make observations of their opponents' actual behavior. The level-K RL solution concept is particularly well-suited to relatively short-term interactions, like the cyber battle analyzed above. However, as interactions draw out over a longer time-frame, we would expect the players to incorporate their opponent's actual behavior into their level-K model of their opponent. One possibility for achieving this type of adaptation is based on a player using a Bayesian variant of fictitious play to set the level 0 distribution of their opponent. In other words, we use the past behavior to update the level 0 distribution of the opponent.

This discussion raises an important question about what happens when the strategic situation is not novel and/or the players have previously interacted. Is the level-K RL model developed here still appropriate? The answer is probably no. In such an *interacted* environment, we should expect the players to have fairly accurate beliefs about each other. Furthermore, these accurate beliefs should lead to well-coordinated play. For example, in the power grid this would mean that the attacker and defender have beliefs that correspond to what the other is actually doing rather than corresponding to some independent model of the other's behavior. In the very least, we should not expect the players to be systematically wrong about each other as they are in the level-K model. Rather, in this interacted environment, player behavior should be somewhere between the completely *non-interacted* level-K models and a full-on equilibrium, such as Nash equilibrium or quantal response equilibrium. The analysis of interacted, one-shot games found in Bono and Wolpert [1, 41] should provide a good starting point for developing a model of an interacted, time-extended game.

Perhaps the most important next step for this work is the process of estimating and validating our model using real data on human behavior. We specifically need data to estimate the parameters of the utility functions and the level K of the players as well as any parameters of their level 0 strategies. After fitting our model to data, we will validate our model against alternative models. The difficult part about choosing alternative models with which to compare our model is that extensive-form games and equilibrium concepts are computationally intractable in the types of do-

mains for which our model is designed. Therefore, feasible alternative models will likely be limited to simplified versions of the corresponding extensive-form game and agent-based simulations of our iterated semi net-form game.

For the smart grid cyber battle analyzed in this paper, there are several options for gathering data. One is to conduct conventional game-theoretic experiments with human subjects in a laboratory setting. Unfortunately, estimating our model, especially with the modeling advances discussed above, will require more data than is practical to collect via such conventional experimental methods which involve actual power grid operators in realistic settings. An alternative method for collecting the large amount of data required is via "crowd-sourcing". In other words, it should be possible to deploy an internet-application version of our smart grid cyber battle to be played by a mixture of undergraduates, researchers, and power engineers. The data from these experiments would then be used to estimate and validate our model.

The methodologies presented here, and the proposed future extensions, also apply to many other scenarios. Among these are several projects related to cyber security as well as the Federal Aviation Administration's NextGen plan for modernizing the National Airspace System. To encompass this range of applications, we are developing libNFG as a code base for implementing and exploring NFGs [24]. The development of this library is ongoing, and modeling advances, like those mentioned above, will be implemented as they become an accepted part of the modeling framework. The libNFG library will ultimately be shared publicly and will enable users to fully customize their own iterated semi net-form game model and choose from a range of available solution concepts and computational approaches.

Acknowledgements This research was supported by the NASA Aviation Safety Program SSAT project, and the Los Alamos National Laboratory LDRD project Optimization and Control Theory for Smart Grid.

References

1. Bono, J., Wolpert, D.H.: Decision-theoretic prediction and policy design of gdp slot auctions. Available at SSRN: `http://ssrn.com/abstract=1815222`. (2011)
2. Brunner, C., Camerer, C.F., Goeree, J.K.: A correction and re-examination of 'stationary concepts for experimental 2x2 games'. American Economic Review (2010)
3. Busoniu, L., Babuska, R., De Schutter, B., Damien, E.: Reinforcement learning and dynamic programming using function approximators. CRC Press (2010)
4. Camerer, C.F.: An experimental test of several generalized utility theories. Journal of Risk and Uncertainty. **2(1)**, 61–104 (1989)
5. Camerer, C.F.: Behavioral game theory: experiments in strategic interaction. Princeton University Press (2003)
6. Camerer, C., Ho, T.H., Chong, J.K.: A cognitive hierarchy model of games. Quarterly Journal of Economics. **119(3)**, 861–898 (2006)
7. Cárdenas, A., Amin, A., Sastry, S.: Research challenges for the security of control systems. Proceedings of the 3rd conference on Hot topics in security. pp. 6:1–6:6, Berkeley, CA, USA, 2008. USENIX Association.

8. Chellapilla, K., Fogel, D.B.: Evolving an expert checkers playing program without using human expertise. IEEE Transactions on Evolutionary Computation. **5(4)**, 422–428 (2001)
9. Costa-Gomes, M., Crawford, V.: Cognition and behavior in two-person guessing games: An experimental study. American Economic Review **96(5)**, 1737–1768 (2006)
10. Costa-Gomes, M.A., Crawford, V.P., Iriberri, N.: Comparing models of strategic thinking in Van Huyck, Battalio, and Beil's coordination games. Journal of the European Economic Association (2009)
11. Crawford, V.P.: Level-k thinking. Plenary lecture. 2007 North American Meeting of the Economic Science Association. Tucson, Arizona (2007)
12. Dayan, P., Balleine, B.W.: Reward, motivation, and reinforcement learning. Neuron. **36(2)**, 285–298 (2002)
13. Fogel, D.B.: Evolutionary computation: Toward a new philosophy of machine intelligence. IEEE Press, 3 edition (2006)
14. Fudenberg, D., Levine, D.K.: The theory of learning in games. MIT Press (1998).
15. Gmytrasiewicz, P. J., Doshi, P.: A framework for sequential planning in multi-agent settings. Journal of Artificial Intelligence Research **24**, 49–79 (2005)
16. Halpern, J.Y., Rego, L.C.: Extensive games with possibly unaware players. `http://www.cs.cornell.edu/home/halpern/papers/aamas06.pdf` Working paper (2007)
17. Harsanyi, J.: Games with Incomplete Information Played by Bayesian Players, I-III. Part I. The Basic Model. Management Science. **14(3)** (1967)
18. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. Journal of Artificial Intelligence Research, **4**, 237–285 (1996)
19. Kagel, J.H., Roth, A.E.: The handbook of experimental economics. Princeton University Press (1995)
20. Kandori, M., Mailath, M., Rob, R.:Learning, mutation, and long run equilibria in games. Econometrica. **61(1)**, 29–53 (1993)
21. Koller, D., Friedman, N.: Probabilistic graphical models: principles and techniques. MIT Press (2009)
22. Kullback, S.:Information theory and statistics. John Wiley and Sons, New York (1959).
23. Kundur, P.: Power system stability and control. McGraw-Hill, New York (1993)
24. Lee, R., Wolpert, D.H.: Game theoretic modeling of pilot behavior during mid-air encounters. Chapter in Decision making with multiple imperfect decision makers. Intelligent Systems Reference Library Series. Springer (2011)
25. Maia, T.: Reinforcement learning, conditioning, and the brain: Successes and challenges. Cognitive, Affective, & Behavioral Neuroscience. 1530-7026. **9(4)**, 343–364. Springer, New York (2009)
26. Maia, T.V., Frank, M.J.: From reinforcement learning models to psychiatric and neurological. Nature Neuroscience. **14**, 154–162 (2011)
27. McKelvey, R., Palfrey, T.R.: Quantal response equilibria for normal form games. Games and Economic Behavior. **10(1)**, 6–38 (1995)
28. McKelvey, R., Palfrey, T.R.: Quantal response equilibria for extensive form games. Experimental Economics. 10.1023/A:1009905800005, **1**, 9–41 (1998).
29. Moriarty, D.E., Schultz, A.C., Grefenstette, J.J.: Evolutionary algorithms for reinforcement learning. The Journal of Artificial Intelligence Research. **11**, 241–276 (1999)
30. Myerson, R.B.: Game theory: Analysis of conflict. Harvard University Press (1997)
31. Nagel, R.: Unraveling in guessing games: An experimental study. The American Economic Review, **85(5)**, 1313–1326 (1995)
32. Plott, C.R., Smith, V.L.: The handbook of experimental economics. North-Holland, Oxford, UK (2008)
33. Robert, C.P., Casella, G.: Monte Carlo statistical methods 2nd ed. Springer (2004)
34. Rummery, G.A., Niranja, M.: Online Q-learning using connectionist systems. Technical report CUED/F-INFENG/TR 166. Engineering department, Cambridge University (1994)
35. Simon, H.A.: Rational choice and the structure of the environment. Psychological Review. **63**(2), 129–138 (1956)

36. Singh, S.P., Jaakkola, T., Jordan, M.I.: Learning without state-estimation in partially observable Markovian decision problems. Proceedings of the Eleventh International Conference on Machine Learning, pp. 284-292, San Francisco (1994)
37. Stahl, D.O., Wilson, P.W.: On players' models of other players: Theory and experimental evidence. Games and Economic Behavior, **10(1)**, 218–254 (1995)
38. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT Press (1998)
39. Tomsovic, K., Bakken, D.E., Venkatasubramanian, V., Bose, A.: Designing the next generation of real-time control, communication, and computations for large power systems. Proceedings of the IEEE **93(5)**, 965–979 (2005)
40. Turitsyn, K., Sulc, P., Backhaus, S., Chertkov, M.: Options for control of reactive power by distributed photovoltaic generators. Proceedings of the IEEE. **99(6)**, 1063–1073 (2011)
41. Wolpert, D.H., Bono, J.W.: Distribution-valued solution concepts. Working paper (2011)
42. Wolpert, D.H.: Unawareness, information theory, and multiagent influence diagrams. Working paper (2012)
43. Wright, J.R., Leyton-Brown, K.: Beyond equilibrium: Predicting human behavior in normal form games. Twenty-Fourth Conference on Artificial Intelligence (AAAI-10) (2010)

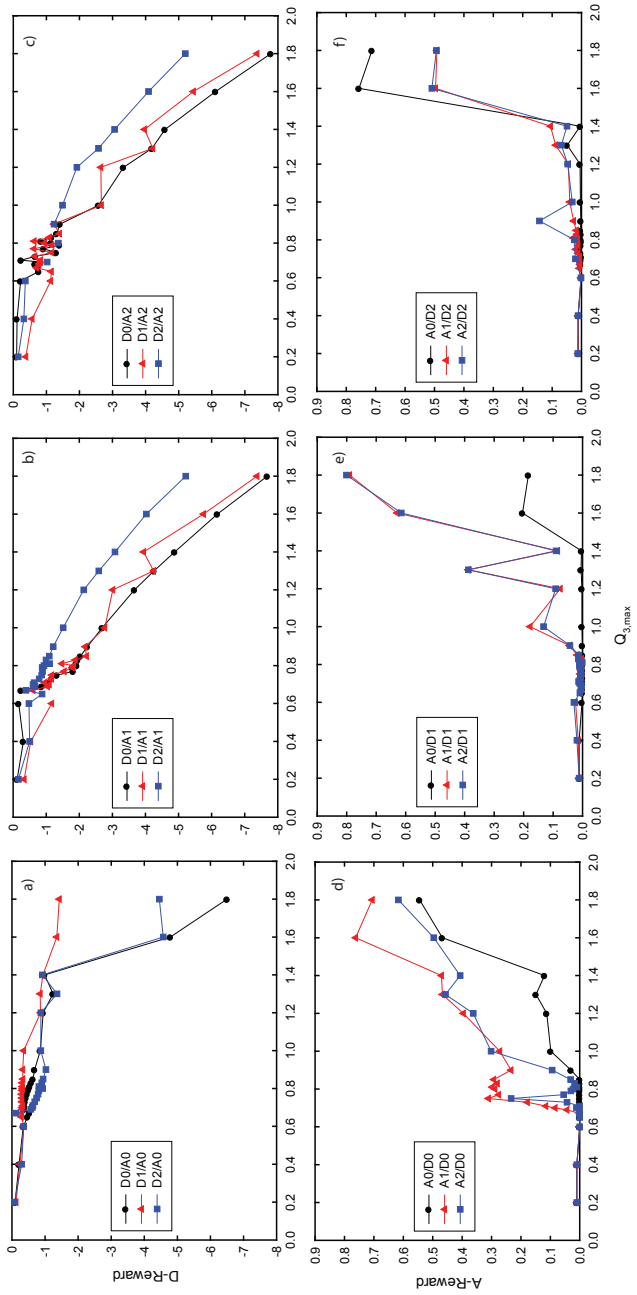

Fig. 4 Average reward per step averaged over 50 episodes as a function of $q_{3,max}$ for all pairings of the defender (D) and attacker (A) through level 2. (a) Reward of D0, D1, and D2 when matched against A0. (b) Same as (a) but for A1. (c) Same as (a) and (b) but for A2. (d) Reward of A0, A1, and A2 when matched against D0. (e) Same as (d) but for D1. (f) Same as (d) and e) but for D2. In general, we observe that as $q_{3,max}$ increases, the defender's average reward decreases and the attacker's average reward increases.

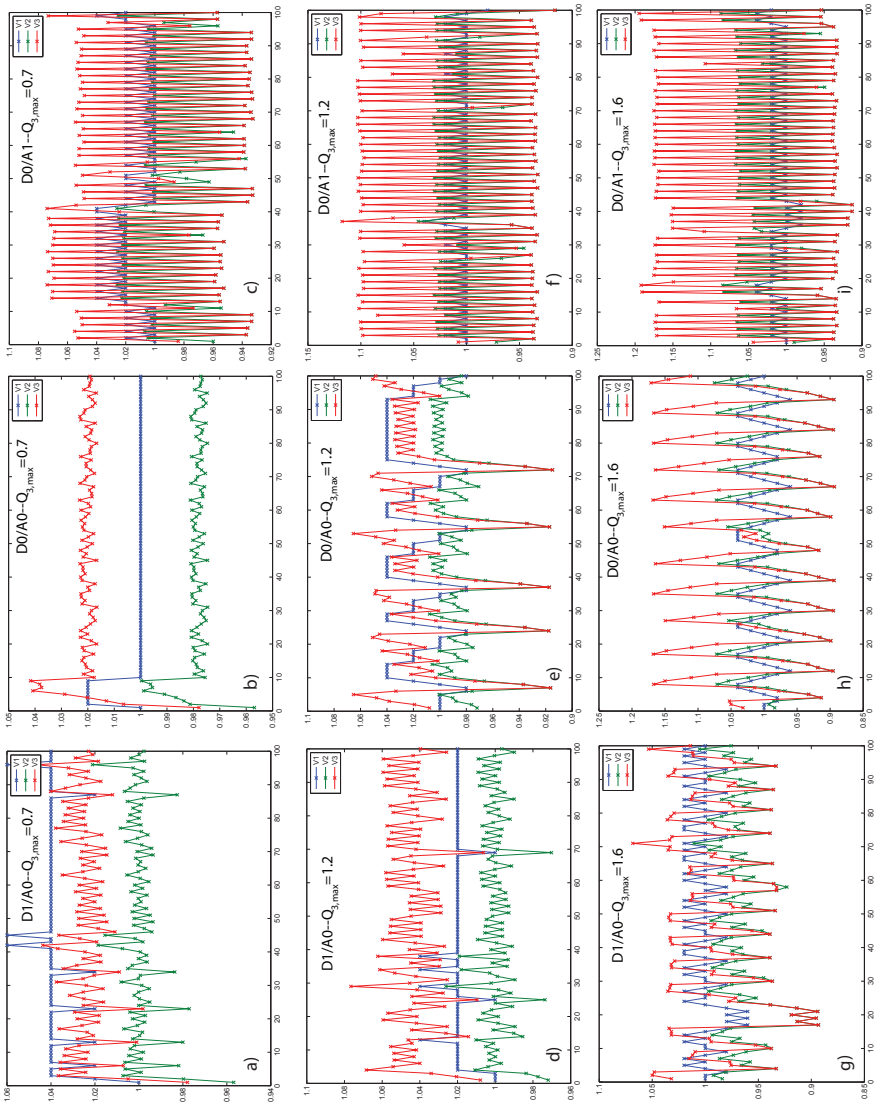

Fig. 5 Simulations of system voltages for level 0 and level 1 that show the evolution in level 1 attacker (A1) and level 1 defender (D1) policies after reinforcement learning training session against their level 0 counterparts D0 and A0. (a) D1 versus A0, (b) D0 versus A0, and (c) D0 versus A1 for $q_{3,max} = 0.7$. (d) D1 versus A0, (e) D0 versus A0, and (f) D0 versus A1 for $q_{3,max} = 1.2$. (g) D1 versus A0, (h) D0 versus A0, and (i) D0 versus A1 for $q_{3,max} = 1.6$. In the center column (D0 versus A0), the attacker becomes increasingly capable of scoring against the defender as $q_{3,max}$ is increased. In the left column (D1 versus A0), the defender is successful at avoiding attacks by not chasing small immediate rewards from voltage centering. In the right column (D0 versus A1), the attacker successfully leverages the level 0 defender's move to help him score.

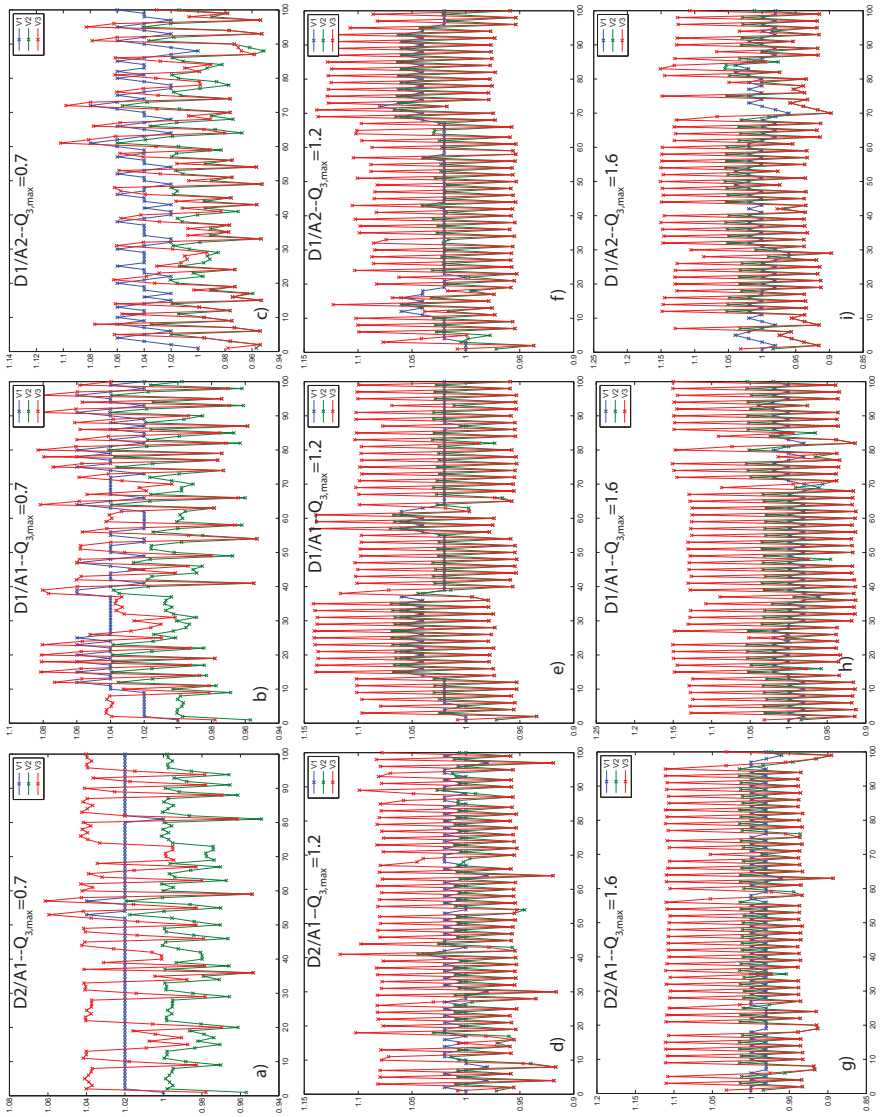

Fig. 6 Simulations of system voltages for level 1 and level 2 that show the evolution in level 2 attacker (A2) and level 2 defender (D2) policies after reinforcement learning training session against their level 1 counterparts D1 and A1. (a) D2 versus A1, (b) D1 versus A1, and (c) D1 versus A2 for $q_{3,max}$ = 0.7. (a) D2 versus A1, (b) D1 versus A1, and (c) D1 versus A2 for $q_{3,max}$ = 1.2. (g) D2 versus A1, (h) D1 versus A1, and (i) D1 versus A2 for $q_{3,max}$ = 1.6.