Reinforcement Learning for Unified Allocation and Patrolling in Signaling Games with Uncertainty

Aravind Venugopal[1], Elizabeth Bondi[2], Harshavardhan Kamarthi[3], Keval Dholakia[1],
Balaraman Ravindran[1], Milind Tambe[2]
[1]Robert Bosch Centre for Data Science and AI, Indian Institute of Technology, Madras
[2]Center for Research on Computation and Society, Harvard University
[3]College of Computing, Georgia Institute of Technology

ABSTRACT

Green Security Games (GSGs) have been successfully used in the protection of valuable resources such as fisheries, forests, and wildlife. Real-world deployment involves both resource allocation and subsequent coordinated patrolling with communication in the presence real-time, uncertain information. Previous game models do not address both of these stages simultaneously. Furthermore, adopting existing solution strategies is difficult since they do not scale well for larger, more complex variants of the game models. We propose a novel GSG model to address these challenges. We also present a novel algorithm, CombSGPO, to compute a defender strategy for this game model. CombSGPO performs policy search over a multidimensional, discrete action space to compute an allocation strategy that is best suited to a best-response patrolling strategy for the defender, learnt by training a multi-agent Deep Q-Network. We show via experiments that CombSGPO converges to better strategies and is more scalable than comparable approaches. From a detailed analysis of the coordination and signaling behavior learnt by CombSGPO, we find that strategic signaling emerges in the final learnt strategy.

KEYWORDS

Reinforcement Learning; Multi-agent Systems; Green Security Games

ACM Reference Format:
Aravind Venugopal, Elizabeth Bondi, Harshavardhan Kamarthi, Keval Dholakia, Balaraman Ravindran, Milind Tambe. 2021. Reinforcement Learning for Unified Allocation and Patrolling in Signaling Games with Uncertainty. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 9 pages.

1 INTRODUCTION

Conservation of natural areas, forests, fisheries and wildlife (“green domains”) is increasingly important in the face of climate change and biodiversity loss. In order to model the protection of these resources, researchers have turned to game theory. Stackelberg Security Games (SSGs) have found extensive use in resource allocation and patrolling in domains such as airport security, transportation, and protection of critical infrastructure [18, 24, 29]. Green Security Games (GSGs), a sub-class of SSGs, have been used to model repeated interactions between defenders and adversaries specifically in green domains. In these domains, defenders protect a finite set of targets (e.g., wildlife) with limited resources, while adversaries plan attacks with knowledge about the defender’s strategy.

Traditional approaches to compute optimal defender strategies for resource allocation or patrolling rely on linear programming (LP) and mixed integer linear programming (MILP) [6, 7, 27]. While a number of such approaches have been developed for GSGs with real-time information [2, 4, 28], these techniques do not scale well with large complex GSGs such as GSG-I [26]. [11] and [12] scale better given their use of deep reinforcement learning algorithms, but they focus only on resource allocation and not on computing patrolling strategies. On the other hand, [26] focuses only on computing patrolling strategies in a game model that involves a single defender and a single adversary. None of [11, 12, 26] consider uncertainty in real-time observation information of the agents or on learning communication strategies between multiple defender agents. Although [4] also incorporates allocation and signaling recommendations for a team of defender resources with uncertain, real-time information, it does not scale effectively to larger areas, multiple adversaries or fine-grained patrols.

We therefore introduce a unified approach for strategic defender allocation and patrolling with inter-resource communication and signaling, given uncertain real-time information. We list our contributions as follows: **(1)** We propose a game model with two stages that captures real-time information through signaling and communication within a team of players; the first stage for handling resource allocation, and the second stage for patrolling once the allocation decision has been made. This is an imperfect-information Extensive Form Game (EFG) capable of representing real-world scenarios with multiple types of defender agents and multiple adversaries. **(2)** We present **Comb**ined **S**ecurity **G**ame **P**olicy **O**ptimization (CombSGPO), a novel solution strategy combining resource allocation with patrolling. Our key algorithmic innovations include: (i) The first use of Competitive Policy Optimization (coPO) [19] in solving GSGs, (ii) Combining action representation learning with coPO to efficiently handle huge, discrete action spaces in GSGs, and (iii) A scalable training scheme combining coPO with multi-agent reinforcement learning, enabling us to solve our novel game model. We also provide extensive experimental results showing the effectiveness of our solution strategy in different types of environments and in the presence of different levels of uncertainty. Our approach ensures very fast convergence to better strategies compared to OptGradFP [11], a fictitious play-based approach and to GUARDSS [4], a LP and MILP-based approach for computing optimal defender strategies. **(3)** Finally, we analyze the signaling behaviors learnt by CombSGPO, illustrating group formation and communication

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

among defender resources, including signaling and notifications. We find that strategic signaling in fact emerges in the final learnt strategy, including a behavior in which drones start signaling or notifying more if they do not see adversaries for some time.

2 PRELIMINARIES

2.1 Green Security Games

GSGs [1, 7], belonging to the larger class of SSGs [14, 23], feature repeated interactions between two players, a leader (defender) and a (boundedly rational) follower (adversary). The two players may consist of multiple individual agents, who may or may not coordinate. A pure strategy is a deterministic mapping from a player's observation space to its strategy space. Throughout this paper, we use the words strategy and policy interchangeably. Each player can play a pure strategy or a mixed strategy, a probability distribution over pure strategies. The leader first commits to a strategy, and then the follower optimizes its reward given the leader's strategy. GSGs can be zero-sum or non-zero-sum.

2.2 Competitive Policy Optimization

In a zero-sum game with two players, the policy gradient algorithm [22] derives policy updates for each agent by maximizing (or minimizing) the linear approximation of the game objective. This does not take the interaction between players into account and therefore, updates the policy of each agent assuming that the other agent is stationary. This generally leads to poor results and non-convergence to Nash Equilibrium (NE).

In contrast, coPO derives policy updates for each agent by computing the NE of the bilinear approximation of the game objective:

$$\theta^1 \leftarrow \theta^1 + \underset{\Delta\theta^1 : \Delta\theta^1 + \theta^1}{\arg\max} \; {\Delta\theta^1}^\top D_{\theta^1}\eta + {\Delta\theta^1}^\top D_{\theta^1\theta^2}\eta\Delta\theta^2 - \frac{1}{2\alpha} \parallel \Delta\theta^1 \parallel^2 \tag{1}$$

$$\theta^2 \leftarrow \theta^2 + \underset{\Delta\theta^2 : \Delta\theta^2 + \theta^2}{\arg\min} \; {\Delta\theta^2}^\top D_{\theta^2}\eta + {\Delta\theta^2}^\top D_{\theta^2\theta^1}\eta\Delta\theta^1 + \frac{1}{2\alpha} \parallel \Delta\theta^2 \parallel^2 \tag{2}$$

where $\top$ means transpose; η represents the game objective, which is the expected utility given the players' policies, parameterized by θ^1 and θ^2 for players 1 and 2, respectively; $D_{\theta^i}\eta$ and $D_{\theta^i\theta^j}\eta$ represent the first order derivative and mixed second order derivative of η, with respect to θ^i and θ^j; α represents the stepsize. As a result of these updates, each agent updates its policy considering what its opponent's move will be at the current timestep and in the future. In a two-player, zero-sum game, this greatly reduces non-stationarity in the environment and leads to stable convergence to NE.

3 GAME MODEL

Our environment is a gridworld, which represents a protected area where, e.g., wildlife, must be protected from illegal poaching. Each cell in the gridworld represents a region in this area with a certain natural resource, e.g., an animal density. We will use animal density throughout the remainder of the paper, but this could also apply to tree density in the case of logging, for example. Details about animal density calculation are provided in Supplementary Material. The defender has n resources to protect the area, of which n_r are rangers (human patrollers) and n_ρ are conservation drones. There are n_a adversaries. We collectively refer to the drones, rangers and adversaries as agents.

Approaching the problem of combined allocation and patrolling as a single large game leads to an acutely deep game tree, even more complex than that of GSG-I introduced in [26]. By using a game model with two stages, we essentially break up the complex problem of combined allocation and patrolling into two stages which can be solved separately. The game is played out in two stages (illustrated in Fig.1) as an EFG featuring sequential interactions between the agents, starting with resource allocation followed by patrolling:

1. **Allocation Stage:** The defender allocates n_ρ drones and n_r rangers to different cells in the environment, which represent the initial locations from which they should start patrolling. In response, n_a adversary allocations are made to target cells from which to start attacking. We assume the adversary has knowledge of the defender's mixed strategy for allocation. We formulate this stage as a single step, two-player, zero-sum game between the defender and adversary (n_a adversary agents collectively) where the payoff from choosing any allocation is equal to the payoff from the patrolling stage that follows it, as we explain next.
2. **Patrolling Stage:** The defender agents (drones and rangers) execute a coordinated patrolling strategy to protect targets, while the adversaries execute a heuristic strategy based on knowledge of the allocations of the defender and the animal densities. Adversaries do not specifically know whether each of these allocations contains a drone or a ranger. This stage ends and the game is terminated when the adversaries flee the grid, are captured, or when the time exceeds T steps, whichever comes first. We formulate this stage as a Partially Observable Stochastic Game (POSG) [9].

In the patrolling stage, each drone is equipped with an object detector [3] that detects an adversary if it is in the current cell of the drone. Drones cannot apprehend an adversary, but can either notify a ranger of the detection and current position, or send a warning signal to an adversary (e.g., turning on lights) to deter an attack and lead him to flee. A drone can warn irrespective of whether the adversary is detected by it or not. An adversary does not do damage to the cells he visits while fleeing, and has fled successfully once he reaches a cell on any edge of the grid. While an adversary can observe a drone signal, he cannot observe notifications sent by drones to rangers. Rangers can apprehend an adversary if they are in the same cell, including while the adversary flees.

We incorporate uncertainty in real-time information during the patrolling stage. We focus on two types of uncertainties: detection and observational. Detection uncertainty is due to uncertainty in object detection [3], e.g., due to occlusion by trees. As in [4], we consider false negative detections specifically, which we denote by β. This means the drones may not always observe an adversary in the same cell. Observational uncertainty is on the side of the adversary, where a drone signals to the adversary but the adversary might not see it, e.g., signals may be blocked by trees. As a result, an adversary may not behave as expected and might not be deterred even by signals from drones. We denote this uncertainty, which is due to false negative observations by the adversary, by κ.

Finally, during patrolling, each defender agent can only observe its current cell, and thus has only local observations, such that even if the observations of all defender agents were pooled together, the environment would still be only collectively partially observable [20]. We also assume that agents can keep track of the cells they have visited from the start of a patrolling episode, which progresses in discrete timesteps. An agent can move only to one of its neighbouring cells at each timestep (and drones may also signal or notify simultaneously). Agent action spaces are as follows:

- Drone (A_ρ) : *[up, down, left, right, stay]* × *[Signal Adversary, Notify Ranger, NoOp]*
- Ranger (A_r) : *[up, down, left, right, stay]*
- Adversary (A_a) : *[up, down, left, right, stay]*

Thus, the patrolling stage POSG consists of S, the set of states; $A \rightarrow A_r \times A_\rho \times A_a$, the set of all actions; $O \rightarrow O_r \times O_\rho \times O_a$, the set of all observations where $O_r \rightarrow O_r^1 \times ... \times O_r^{n_r}$ is the set of observations of all rangers, $O_\rho \rightarrow O_\rho^1 \times ... \times O_\rho^{n_\rho}$ is the set of observations of all drones, $O_a \rightarrow O_a^1 \times ... \times O_a^{n_a}$ is the set of observations of all adversaries; $\mathcal{T} \rightarrow S \times A$, the state transition function; $R : S \times A \rightarrow \mathcal{R}$, the reward function; discount factor $\gamma \in [0, 1]$.

Now, we describe the rewards. At each timestep in the patrolling stage, defender agents collectively receive a positive reward r^+ if an adversary is caught, and a zero reward if the adversary flees. If the adversary is not caught or deterred in the current timestep, the defender agents receive a penalty r^- for each target attacked by each adversary, which is proportional to the animal density of the attacked target. In addition to this, a drone agent can receive r^c for notifying or signaling when it detects an adversary and $\bar{r^c}$, a penalty for notifying or signaling without detecting an adversary. We keep the magnitudes of these additional individual rewards much lower than the collective rewards, as they are meant to ensure efficient communication by penalizing false and redundant notifications or signaling. The sum of rewards of all defender agents at any timestep t is given by r_t^d. We denote drone, ranger, and adversary strategies for the patrolling stage by ϕ^ρ, ϕ^r and ϕ^a, respectively.

In the allocation stage, the payoff to the defender from choosing an allocation is R^d, given by $\sum_{t=0}^{T} r_t^d$, which is the cumulative reward from playing out the patrolling stage from that allocation. When the adversary chooses an allocation, it receives a payoff of R^a, where R^a = -R^d. We denote defender and adversary allocation strategies by π^d and π^a, respectively.

4 METHODOLOGY

We introduce our algorithm, CombSGPO, for computing the optimal defender strategy for the two-stage game model introduced in Sec. 3. CombSGPO first computes a defender patrolling strategy for the patrolling stage of our game and then chooses defender allocations that are best suited for this patrolling strategy. Throughout this section, "allocation" refers to the cells to which defender or adversary agents are deployed during the allocation stage. In this section, we first describe the behavioral model we use for our adversary. We then describe how we compute defender strategies for the patrolling and allocation stages, and introduce CombSGPO, which combines the patrolling and allocation strategy computations.

4.1 Adversary Behavioral Model

The adversary allocation strategy, π^a, is parameterized by the weights of a neural network, which are updated to maximize the best response against the defender's allocation strategy. Adversary strategy for the patrolling stage, ϕ^a, is a heuristic strategy based on the animal density of each cell in the grid and the distance of the adversary from defender agent allocations. This is meant to reflect that adversaries prefer to attack regions with more animals, while trying to be as distant from the defender agents as possible. An adversary ranks each cell based on its minimum distance from the allocation of any defender agent, such that cells that are farther away from defender agents are given higher ranks. It then assigns a score to each cell, equal to the average of the animal density and the normalized distance rank of that cell. Over subsequent episodes, the score for a cell is updated:

$$score_{av} = 0.5ad + 0.5dr \tag{3}$$

$$score_{ep+1} = score_{ep} + 0.1(score_{av} - score_{ep}) \tag{4}$$

where ad, dr, and $score_{av}$ are the animal density, normalized distance rank, and average score of each cell. At each timestep, an adversary chooses the neighbouring cell with the highest score to attack next.

4.2 Computing Patrolling Strategy

To compute coordinated patrolling strategies with communication for a team of n_r rangers and n_ρ drones, we use a multi-agent Double Deep Q Network (DDQN) [25].

We use the Centralized Training and Decentralised Execution (CTDE) [16] framework for training the multi-agent DDQN. The patrolling strategy for each drone i, $\phi^i_{\theta^\rho}$: O^i_ρ x $A_\rho \rightarrow$ [0, 1] is learnt using a DDQN, where θ^ρ represents the parameters of the DDQN shared by all drones. Similarly, a DDQN learns the patrolling strategy for each ranger j, $\phi^j_{\theta^r}$: O^j_r x $A_r \rightarrow$ [0, 1] with parameters θ^r shared by all rangers. We thus use a centralized controller for all drones and a centralized controller for all rangers, sharing information with each other through the communication actions of the drones. While each DDQN learns from the experiences of all the drones/rangers that it controls (centralized learning), actions for each drone/ranger are taken independently (decentralized execution).

The state representation used as input to the DDQN is a 3D Tensor with 9 channels, each with the same width and height as the grid: (1) The current position of the drone/ranger in the grid, (2) Whether an adversary has been detected in the current cell, (3) Positions of all other drones, (4) Positions of all other rangers, (5) The outputs of the object detectors on each drone (0 or 1), (6) Whether each drone is currently notifying rangers or not, (7) Whether each drone is currently signaling or not, (8) Animal density of the visited cell, and (9) Visitation counts of each defender agent in the grid since the beginning of the episode. More details about the neural network architecture are described in Supplementary Material.

4.3 Computing Allocation Strategy

Because we formulate the allocation stage as a two-player, zero-sum game, the Stackelberg Security Equilibrium (SSE) strategy for the allocation stage is the same as a NE strategy [8]. Each

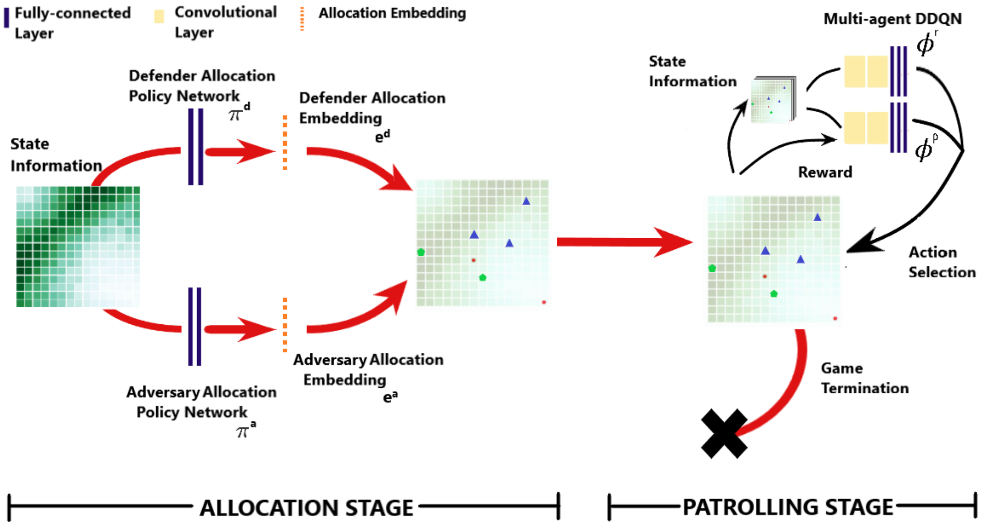

Figure 1: CombSGPO algorithm. (The state information in the allocation stage represents the animal density distribution. Green pentagons: Rangers, Blue triangles: Drones, Red dots: Adversaries)

defender allocation consists of initial cells for $(n_r + n_\rho)$ defender agents, while each adversary allocation consists of initial cells for n_a adversaries. For a grid with N cells, there are $\frac{N!}{(n_\rho+n_r)!n_\rho!n_r!}$ possible pure strategy allocations for the defender. Computing a NE allocation strategy would involve searching in a huge, discrete and multi-dimensional space of all possible allocation strategies. For huge values of N, this computation quickly becomes intractable using traditional approaches based on LP and MILP. We therefore propose a gradient-based approximation using the policy gradient theorem and coPO.

However, the vanilla policy gradient algorithm [22] performs poorly with large discrete action spaces. To address this, we can instead decompose a policy into a component that acts in a latent space of action representations (embeddings) and a component that transforms these representations into actual actions, as shown in [5]. This allows generalization over allocations, as similar allocations have similar action representations, and improves performance, while speeding up learning. We therefore use learnt embeddings to represent allocations.

We generate 100K allocations for the defender and adversary, sampled from $\mathcal{D}_d$ and $\mathcal{D}_a$, the set of possible allocations for each player and train autoencoder neural networks f^d and f^a to learn embeddings for these allocations. The network consists of two fully-connected dense layers with a *tanh* activation after the first layer. Once training is complete, we use the encoder part of the network to obtain embeddings e^d and e^a of sizes k_d and k_a for the defender and adversary.

The defender and adversary allocation policies, are π^d, parameterized by $\mathbf{w}^d$ and π^a, parameterized by $\mathbf{w}^a$; where $\mathbf{w}^d$ and $\mathbf{w}^a$ represent the weights of neural networks. π^d and π^a search over the space of all possible e^d and e^a, respectively. Details about neural network architecture are provided in Supplementary Material. Each network outputs a mixed strategy over embeddings, out of which one is sampled. The sampled embeddings are then mapped back to the allocation that they represent most closely, and the patrolling stage is simulated to receive payoffs R^d and R^a. The animal density distribution of the grid, s, represented by a 2D tensor, is fed as state to π^d and π^a. The utilities of the defender and adversary (U^d and U^a = -U^d) are the expected rewards, given π^d and π^a:

$$U^d(\mathbf{w}^d, \mathbf{w}^a) = \mathbf{E}_{s,e^d,e^a}[R^d(s, e^d, e^a)]$$
$$= \int_s \int_{e^d} \int_{e^a} \mathbf{P}(s)\pi^d(e^d|s; \mathbf{w}^d)\pi^a(e^a|s; \mathbf{w}^a)R^d(s, e^d, e^a)\, ds\, de^d\, de^a \tag{5}$$

We must then compute the best-response weights, $\mathbf{w}^{d*}$ and $\mathbf{w}^{a*}$:

$$\mathbf{w}^{d*} \epsilon \arg\max_{\mathbf{w}^d} \min_{\mathbf{w}^a} U^d(\mathbf{w}^d, \mathbf{w}^a) \tag{6}$$

$$\mathbf{w}^{a*} \epsilon \arg\min_{\mathbf{w}^a} U^d(\mathbf{w}^{d*}, \mathbf{w}^a) \tag{7}$$

The weights are updated by coPO [19] to arrive at an approximate NE for $\mathbf{w}^{d*}$ and $\mathbf{w}^{a*}$, as follows:

$$\begin{aligned} \mathbf{w}^d \leftarrow \mathbf{w}^d + \underset{\Delta\mathbf{w}^d:\Delta\mathbf{w}^d+\mathbf{w}^d}{\arg\max}\ & \Delta\mathbf{w}^{d\top} D_{\mathbf{w}^d} U^d + \\ & \Delta\mathbf{w}^{d\top} D_{\mathbf{w}^d\mathbf{w}^a} U^d \Delta\mathbf{w}^a - \frac{1}{2\alpha} \parallel \Delta\mathbf{w}^d \parallel^2 \end{aligned} \tag{8}$$

$$\begin{aligned} \mathbf{w}^a \leftarrow \mathbf{w}^a + \underset{\Delta\mathbf{w}^a:\Delta\mathbf{w}^a+\mathbf{w}^a}{\arg\min}\ & \Delta\mathbf{w}^{a\top} D_{\mathbf{w}^a} U^d + \\ & \Delta\mathbf{w}^{a\top} D_{\mathbf{w}^a\mathbf{w}^d} U^d \Delta\mathbf{w}^d + \frac{1}{2\alpha} \parallel \Delta\mathbf{w}^a \parallel^2 \end{aligned} \tag{9}$$

where α represents the stepsize.

4.4 CombSGPO Algorithm

We describe CombSGPO, which combines computation of optimal strategies for allocation and patrolling. The training steps are summarized in Algorithm 1. Fig. 1 illustrates the CombSGPO pipeline. First, we learn the defender patrolling strategy, choosing random allocations for the agents at the start of each training episode and simulating the patrolling stage, to converge to patrolling strategies ϕ^{ρ} and ϕ^{r} in lines 1 to 5. Lines 6 and 7 describe how we learn allocation embeddings.

For a given state s of the grid and pre-trained allocation embeddings, our algorithm solves the allocation stage as follows. At each episode, n_b embeddings (e_i^d and e_i^a) are sampled from π^d and π^a, as shown in line 12. Each embedding is then matched to the closest allocation that it represents using a cosine similarity measure to get n_b defender and adversary allocations. These allocations are then used as initial positions for the patrolling stage. The simulator for the patrolling stage then runs n_b patrols to completion, returning payoffs R_i^d and R_i^a for each pair of defender and adversary allocations, as described in line 13. The payoffs obtained are used to update π^d and π^a using coPO in line 15. We use n_b=10 in our experiments.

Self-play based approaches [10, 11, 15] store the game state and sampled actions at each episode in a replay memory. They update each player's policy towards the best response to the other player's average policy using linear approximation during optimization. This leads to large memory requirements and requires a huge number of sampled trajectories to approach equilibrium. Our approach directly updates each player's policy towards an approximate NE by using a bilinear approximation during its optimization step. Therefore, it is more sample efficient and also eliminates the need for using a replay memory.

5 EXPERIMENTS AND RESULTS

We now present several experiments evaluating CombSGPO on the game model in Sec. 3. For the game, we present results for n_{ρ} = 3 drones, n_r = 2 rangers and gridsizes of 15x15 and 10x10. We ran experiments with n_a = 1 as well as n_a = 2, to show how the allocation and patrolling behavior changes when there is just one adversary compared to when there are multiple adversaries, which requires splitting the team into sub-teams. We did this for random as well as spatial animal densities (based on distance to the border and other real-world features like roads and rivers). For convenience, we denote the experiments by: SS (single adversary, spatial animal densities), SR (single adversary, random animal densities), MS (multiple adversaries, spatial animal densities) and MR (multiple adversaries, random animal densities). Since we focused on real-world applicability, we focus primarily on SS and MS experiments in this section performed at different levels of uncertainty (β and κ), while we defer SR and MR results to Supplementary Material. All results are averaged over five runs. Hyperparameters used for training the neural networks are detailed in Supplementary Material. We compared CombSGPO against the following baselines, all of which used a multi-agent DDQN for patrolling:

(1) Random: The defender allocations were assigned at random.
(2) Policy Gradient (PG) [22]: We used this algorithm to make players learn best responses to each other's strategies via policy gradient updates.
(3) OptGradFP [11]: While this also uses policy gradient updates to learn best responses, the Fictitious Play [15] training framework is used to make both players converge to NE.

Algorithm 1: CombSGPO

```
   /* Train the DDQNs for drones and rangers to get
      patrolling strategies                              */
1  repeat
2      Sample a random allocation for defenders and
         adversaries;
3      Run the episode and add experiences to DDQNs' replay
         memory;
4      Update parameters $\theta^{\rho}$ and $\theta^{r}$ for DDQNs for $\phi^{\rho}$ and $\phi^{r}$;
5  until convergence;
   /* Train defender and adversary allocation
      strategies                                         */
6  Sample 100000 random allocations each for defenders and
     adversaries in $\mathcal{D}_d$ and $\mathcal{D}_a$ respectively;
7  Train autoencoders $f_d : \mathcal{D}_d \rightarrow \mathcal{R}^{k_d}$ and $f_a : \mathcal{D}_a \rightarrow \mathcal{R}^{k_a}$;
8  Initialize $\mathbf{w}^d$ and $\mathbf{w}^a$;
9  repeat
10     Obtain allocation stage state = $s$;
11     for $i$ = 1 to $n_b$ do
12         Sample $e_i^d$ from $\pi^d$ and $e_i^a$ from $\pi^a$; Get allocations
             $alloc_i^d = \arg\min_{alloc \in \mathcal{D}_d} ||f_d(alloc) - e_i^d||^2$ and
             $alloc_i^a = \arg\min_{alloc \in \mathcal{D}_a} ||f_a(alloc) - e_i^a||^2$;
13         Simulate patrolling stage with allocations $alloc_i^d$,
             $alloc_i^a$, patrolling stage strategies $\phi^{\rho}$, $\phi^{r}$, $\phi^{a}$ and
             receive rewards $R_i^d$, $R_i^a$;
14     end
15     Update parameters $\mathbf{w}^d$ and $\mathbf{w}^a$ by coPO;
16 until convergence;
```

(4) GUARDSS [4]: We replaced CombSGPO allocation with a GUARDSS allocation, then ran our patrolling strategy for direct utility comparison. Specifically, we used the initial drone, ranger, and adversary locations (no signaling) from all pure strategies in the optimal mixed strategy for the GUARDSS model. GUARDSS only considers a single adversary, so MR and MS are left blank.

First, we evaluated the performance of CombSGPO with different gridsizes, animal densities, and number of adversaries, with no uncertainty (β = 0, κ = 0). We sampled 150 episodes once the policies converged and averaged the defender utilities from these episodes. Allocation stage training curves for SS and MS on a 15x15 grid are shown in Fig. 2. Average utilities for different experiments are shown in Tables 1 and 2. CombSGPO significantly outperformed the baselines, converging faster and towards strategies with greater expected utility for the defender. The variance of rewards in CombSGPO is much lower, indicating robust strategies.

Next, we evaluated how the perfomance of CombSGPO varied with detection and observational uncertainty. We performed experiments with three sets of values for β and κ: [0,0], [0.25,0.25] and [0.75,0.75], to simulate increasing levels of uncertainty. Defender

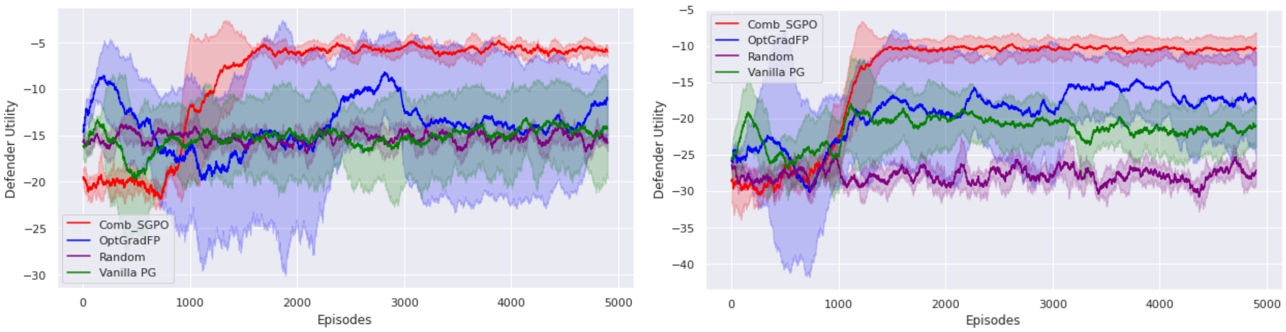

Figure 2: Allocation stage training curves for a 15x15 grid (β=0,κ=0): SS (left) and MS (right). (GUARDSS not trained)

	Random	PG	OptGradFP	GUARDSS	CombSGPO
SR	-22.54	-16.84	-19.72	-24.34	**-9.25**
MR	-39.23	-51.97	-41.55	-	**-24.49**
SS	-15.35	-14.99	-14.49	-14.11	**-5.53**
MS	-27.95	-21.42	-15.41	-	**-10.39**

Table 1: Average defender utilities for a 15x15 grid

	Random	PG	OptGradFP	GUARDSS	CombSGPO
SR	-6.25	-7.12	-5.83	-7.51	**-2.29**
MR	-13.93	-12.51	-16.92	-	**-7.72**
SS	-5.37	-4.55	-3.47	-6.03	**-1.63**
MS	-12.41	-9.69	-7.85	-	**-4.65**

Table 2: Average defender utilities for a 10x10 grid

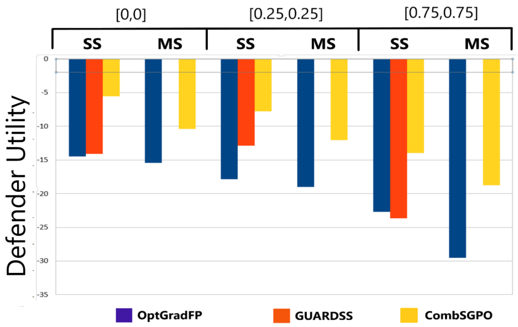

Figure 3: Average defender utilities under varying [β, κ]

utilities were calculated from 150 episodes sampled once the policies converged for SS and MS on a 15x15 grid. Only OptGradFP and GUARDSS have been considered, as the other baselines did not converge to an equilibrium. Results are shown in Fig. 3, with CombSGPO having the best performance, as it is most positive. In general, defender utility decreases with an increase in uncertainty. We also compared the patrolling strategy learnt by our algorithm against MADDPG (Multi Agent Deep Deterministic Policy Gradient)[16] in Supplementary Material.

For a 15x15 grid with SS, we show heatmaps of target cells attacked by sampling 100 adversary allocations and simulating the game, from the final strategies of the adversary for each algorithm, in Fig. 4. This is a direct measure of how effective the final learnt defender strategies of each algorithm are, as better defender strategies limit attacks to smaller regions of the grid and regions with lower animal densities. The animal density distribution is also shown alongside for comparison. In all results, there was a clear tendency for the adversary to attack cells that were closer to the boundaries of the grid. While adversaries in both OptGradFP and PG attacked multiple quadrants and larger regions overall, both CombSGPO and GUARDSS defenders were able to limit attacks mainly to one quadrant, with CombSGPO reducing attacks to a smaller region.

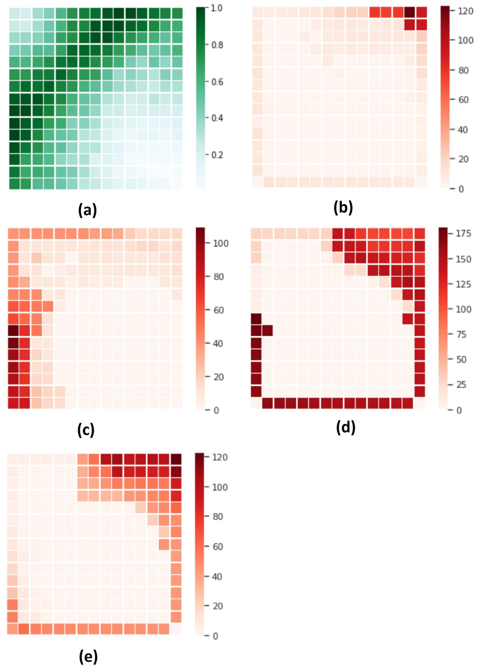

Figure 4: Animal density (a) & heatmaps of sampled attacks for: (b) CombSGPO, (c) GUARDSS, (d) OptGradFP, (e) PG.

Having shown that CombSGPO outperformed all considered baselines quantitatively in terms of faster convergence, lesser variance, greater defender utilities, and qualitatively in terms of greater

Gridsize	OptGradFP	GUARDSS	CombSGPO
10x10	11.11±0.006	65.09±15.45	**4.19±0.001**
15x15	24.62±0.029	79.60±20.65	**9.92±0.008**

Table 3: Timing results (minutes) for 15x15 and 10x10 grids

protection to target cells, we evaluated the time taken to converge to equilibrium for CombSGPO, OptGradFP, and GUARDSS on a single core of Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90GHz. Table 3 shows the results for SS on 15x15 and 10x10 grids, averaged over 5 runs, with CombSGPO performing best. We maintained a time-out period of 60 minutes in all experiments. GUARDSS timed out in 4 out of 5 runs with the 10x10 grid and 2 out of 5 runs with the 15x15 grid.

6 ANALYSIS OF POLICIES LEARNT BY COMBSGPO

In this section, we provide a detailed qualitative analysis of the final strategies learnt by CombSGPO, over multiple β and κ. For better insight, we show snapshots taken from sampled episodes, illustrating interesting strategic behavior. In all snapshots, red dots represent adversaries, green pentagons represent rangers, blue triangles represent drones, and grey/yellow triangles represent signaling/notifying drones, respectively. Signaling and notifying actions are emphasized with a black box around the snapshot. The animal density of each cell is shown by the shade; darker shades have higher animal density.

Throughout all experiments, we observed that CombSGPO learnt to deploy all drones together in one group and all rangers together in another group, as this allowed a greater area to be patrolled by the defender agents. We also observed that *often, when a ranger was too far away from the drones to apprehend an adversary if the drone notified her, the drones learnt to signal the adversary and make him flee instead of notifying*, similar to the strategic signaling modeled by GUARDSS. This is illustrated in Fig. 5, which shows a sampled SS episode under β=0, κ=0. At t=0, the drones and rangers are deployed as separate groups, with the adversary deployed closer to the drones. At t=7, we can see the drones moving closer to the adversary, while signaling at the same time. The arrow indicates that the rangers are far away from the adversary. The next frame at t=8 shows that the adversary is signalled by a drone in the same cell (indicated by a star symbol), causing him to flee.

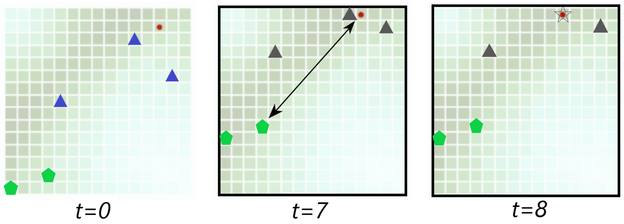

Figure 5: Sampled SS episode(β=0,κ=0).

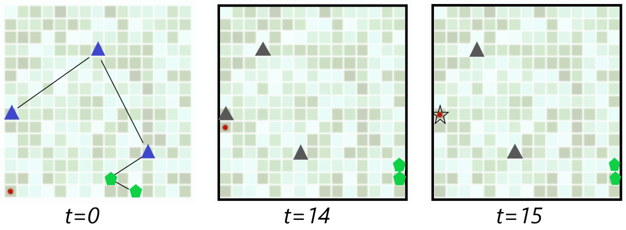

Figure 6: Sampled SR episode(β=0,κ=0).

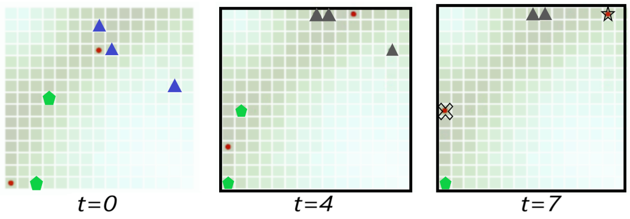

Figure 7: Sampled MS episode(β=0.25,κ=0.25).

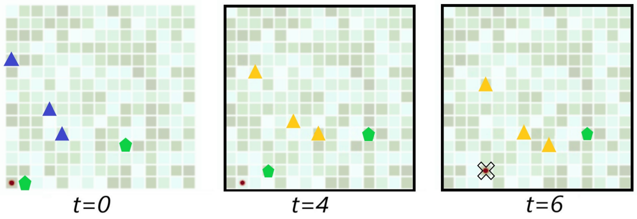

Figure 8: Sampled SR episode(β=0.75,κ=0.75).

In some cases, we observed that defender agents allocated in a web-like manner around the adversary and closed in on the adversary, cornering him. The attractiveness of a cell to an adversary also depends on his distance from initial allocations of defender agents (as described in Sec. 4.1). Through this allocation pattern, *the adversary was deliberately forced away from the areas of greater animal density and towards the corners and edges.* Fig 6 shows such a pattern for a sampled SR episode under β=0, κ=0, where defender agents deploy in a web-like formation (indicated by the black lines).

The allocation and signaling strategies were even clearer in the MR and MS experiments, as illustrated in Fig. 7, a sampled MS episode under β=0.25, κ=0.25. While one adversary is caught by the rangers (capture indicated by X), the other one, deployed in a completely different region, is forced to flee by the drones.

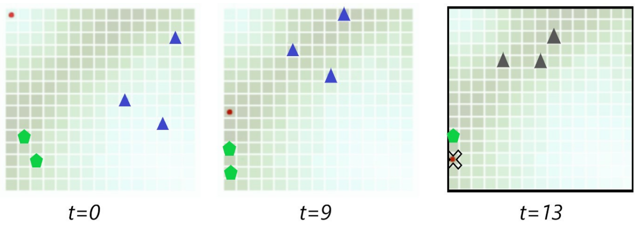

Figure 9: Sampled SS episode(β=0.25,κ=0.25).

Apart from signaling adversaries, notifications from drones to rangers were also observed. Interestingly, in a lot of cases where the adversary was allocated closer to the rangers than the drones, we saw spurts of notification activity from the drones in the steps just before the adversary was caught, as illustrated in Fig. 8, a sampled SR episode under β=0.75, κ=0.75. This may indicate *strategic notifications* to suggest to the rangers that adversaries are probably closer to them, rather than the drones' current region.

In addition to this, *whenever the drones failed to detect the adversary even after many timesteps of patrolling, they responded with signaling and notifications.* In particular, they went into a state of constant signaling activity in the SS, MS, and MR experiments, and into a state of constant notification activity in SR experiments. Fig. 9 shows a sampled SS episode β=0.25, κ=0.25 where the drones signal after 13 timesteps. This is similar in spirit to the ranger moving in the GUARDSS model when an adversary is not seen for some time. In our model, this could also indicate that drones are trying to compensate for possible false negative detections, or possibly that drones are suggesting that adversaries are near rangers instead.

As uncertainty increases from β=0, κ=0 to β=0.75, κ=0.75, we see a marked decrease in the amount of signaling and notification activity in SR experiments while the amount of signaling and notification activity stays the same in SS, MS, and MR experiments.

7 RELATED WORK

In RL for GSGs, [11, 12] use policy gradient learning for resource allocation, and [26] uses an algorithm based on Deep Q-Learning [17] for computing patrolling strategies with the starting cell chosen randomly from a set of four possible allocations. While [26] models real-time information, it does not incorporate uncertainty, signaling, or multiple defender/adversary agents. [13] also uses RL for GSGs, but with a model-based approach and assuming periodic observability of the adversary's location.

Our work is also similar to previous models for sequential team decision-making under uncertainty. Earlier [21] presented a model of sequential decision making of teams of agents against an adversary, combining security games with dec-MDPs. There is a rich literature of algorithms for dec-MDPs, including early work on distributed POMDPs to model multiagent teamwork [20]. [21] did not focus on signaling games, which bring in significant complexities, and limited scale up of 10 targets. This paper explores signaling games with a complementary approach using RL, with a significant scale up allowing for a 15x15 grid.

[4] also models uncertainty in real-time information in an application that uses conservation drones to prevent illegal wildlife poaching. Drones fitted with an object detection system called SPOT [3] provide (uncertain) automatic detection of potential poachers (adversaries) and animals. If suspicious activity is detected, nearby park rangers (defenders) can be mobilized to respond. Warning signals may also be deployed by the drones to deter poachers. Signaling can be done deceptively in order to lead poachers to believe that park rangers may respond, even when rangers are unable to do so. While this approach models allocation and signaling decisions, it does not model patrolling strategies or multiple adversaries, and is not efficiently scalable to larger protected areas.

In contrast, our work proposes a unified solution to resource allocation and patrolling. We also focus on strategic signaling and communication between agents for coordinated patrolling in the presence of uncertain, real-time information, which is an area that is yet to be addressed in complex GSGs by using RL methods.

8 DISCUSSION AND CONCLUSION

We introduce a novel solution approach based on reinforcement learning for computing defender strategies in a two-stage GSG, that combines resource allocation, patrolling, communication between resources, and signaling in the presence of uncertain real-time information. We showed that our approach outperforms comparable approaches over different types of environments that simulate green security domains, including protection of animals from wildlife poaching. We also showed that our trained models learn strategic behavior such as formation of sub-teams within a team of drones and rangers, coordinated patrolling formations, and strategic signaling in the presence of uncertainty to ward off and/or apprehend poachers.

An interesting future direction would be the replacement of the binary object detectors with real-time image input from object detectors on drone cameras. This would introduce intrinsic uncertainty into the model and also take us a step closer to deploying such an application in real-world scenarios.

Broader Impacts: The idea for strategic signaling came from discussions with domain experts at Air Shepherd (https://airshepherd.org/), who tried illuminating the lights on conservation drones to deter poachers. However, to our knowledge, no strategic signaling system has been deployed. Our system could aid in deployment by providing fine-grained patrolling suggestions and strategies to minimize errors due to uncertainty. CombSGPO depends on knowledge of an underlying animal density and on assumptions of certain knowledge by poachers, such as the poacher knowing an initial allocation in the patrolling stage. If a poacher knew the defender's location at all times, he may be able to evade the rangers, and a better strategy for the defender may be to simply protect high-valued targets. We also assume that poachers know that signaling implies rangers are responding to prevent poaching from occurring. In the real world, it is possible that poachers would not know this is the case and may fear worse, such as the drone being armed. There may also be adverse effects of signaling on animals. Any deployment must therefore be done with transparency and collaboration with local stakeholders, especially including traditionally marginalized communities. Depending on this local context, other interventions may be better, or may be used together with strategic signaling and CombSGPO.

9 ACKNOWLEDGEMENTS

This research was partly sponsored by the Army Research Office and was accomplished under Grant Number: W911NF-17-1-0370. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

REFERENCES

[1] Anjon Basak, Fei Fang, Thanh Hong Nguyen, and Christopher Kiekintveld. 2016. Combining graph contraction and strategy generation for green security games. In *International Conference on Decision and Game Theory for Security*. Springer, 251–271.

[2] Nicola Basilico, Giuseppe De Nittis, and Nicola Gatti. 2015. A security game model for environment protection in the presence of an alarm system. In *International Conference on Decision and Game Theory for Security*. Springer, 192–207.

[3] Elizabeth Bondi, Fei Fang, Mark Hamilton, Debarun Kar, Donnabell Dmello, Jongmoo Choi, Robert Hannaford, Arvind Iyer, Lucas Joppa, Milind Tambe, et al. 2018. Spot poachers in action: Augmenting conservation drones with automatic detection in near real time. In *IAAI*. 7741–7746.

[4] Elizabeth Bondi, Hoon Oh, Haifeng Xu, Fei Fang, Bistra Dilkina, and Milind Tambe. 2020. To Signal or Not To Signal: Exploiting Uncertain Real-Time Information in Signaling Games for Security and Sustainability. In *AAAI*. 1369–1377.

[5] Yash Chandak, Georgios Theocharous, James Kostas, Scott Jordan, and Philip S Thomas. 2019. Learning action representations for reinforcement learning. *arXiv preprint arXiv:1902.00183* (2019).

[6] Fei Fang, Thanh Hong Nguyen, Rob Pickles, Wai Y Lam, Gopalasamy R Clements, Bo An, Amandeep Singh, Milind Tambe, Andrew Lemieux, et al. 2016. Deploying PAWS: Field Optimization of the Protection Assistant for Wildlife Security. In *AAAI*, Vol. 16. 3966–3973.

[7] Fei Fang, Peter Stone, and Milind Tambe. 2015. When Security Games Go Green: Designing Defender Strategies to Prevent Poaching and Illegal Fishing. In *IJCAI*. 2589–2595.

[8] Drew Fudenberg, Fudenberg Drew, David K Levine, and David K Levine. 1998. *The theory of learning in games*. Vol. 2. MIT press.

[9] Eric A Hansen, Daniel S Bernstein, and Shlomo Zilberstein. 2004. Dynamic programming for partially observable stochastic games. In *AAAI*, Vol. 4. 709–715.

[10] Johannes Heinrich and David Silver. 2016. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121* (2016).

[11] Nitin Kamra, Umang Gupta, Fei Fang, Yan Liu, and Milind Tambe. 2018. Policy Learning for Continuous Space Security Games Using Neural Networks. In *AAAI*. 1103–1112.

[12] Nitin Kamra, Umang Gupta, Kai Wang, Fei Fang, Yan Liu, and Milind Tambe. 2019. DeepFP for Finding Nash Equilibrium in Continuous Action Spaces. In *International Conference on Decision and Game Theory for Security*. Springer, 238–258.

[13] Richard Klima, Karl Tuyls, and Frans A Oliehoek. 2018. Model-based reinforcement learning under periodical observability. In *2018 AAAI Spring Symposium Series*.

[14] Dmytro Korzhyk, Zhengyu Yin, Christopher Kiekintveld, Vincent Conitzer, and Milind Tambe. 2011. Stackelberg vs. Nash in security games: An extended investigation of interchangeability, equivalence, and uniqueness. *Journal of Artificial Intelligence Research* 41 (2011), 297–327.

[15] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. 2017. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in neural information processing systems*. 4190–4203.

[16] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275* (2017).

[17] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[18] James Pita, Manish Jain, C Western, P Paruchuri, J Marecki, M Tambe, F Ordonez, and Sarit Kraus. 2009. Armor software: A game theoretic approach to airport security. *Protecting Airline Passengers in the Age of Terrorism* (2009), 163.

[19] Manish Prajapat, Kamyar Azizzadenesheli, Alexander Liniger, Yisong Yue, and Anima Anandkumar. 2020. Competitive Policy Optimization. *arXiv preprint arXiv:2006.10611* (2020).

[20] David V Pynadath and Milind Tambe. 2002. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of artificial intelligence research* 16 (2002), 389–423.

[21] Eric Shieh, Albert Jiang, Amulya Yadav, Pradeep Reddy VARAKANTHAM, and Milind Tambe. 2014. Unleashing dec-mdps in security games: Enabling effective defender teamwork. (2014).

[22] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*. 1057–1063.

[23] Milind Tambe. 2011. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge university press.

[24] Jason Tsai, Shyamsunder Rathi, Christopher Kiekintveld, Fernando Ordonez, and Milind Tambe. 2009. IRIS-a tool for strategic security allocation in transportation networks. *AAMAS (Industry Track)* (2009), 37–44.

[25] Hado Van Hasselt, Arthur Guez, and David Silver. 2015. Deep reinforcement learning with double q-learning. *arXiv preprint arXiv:1509.06461* (2015).

[26] Yufei Wang, Zheyuan Ryan Shi, Lantao Yu, Yi Wu, Rohit Singh, Lucas Joppa, and Fei Fang. 2019. Deep reinforcement learning for green security games with real-time information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 1401–1408.

[27] Haifeng Xu, Benjamin Ford, Fei Fang, Bistra Dilkina, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, et al. 2017. Optimal patrol planning for green security games with black-box attackers. In *International Conference on Decision and Game Theory for Security*. Springer, 458–477.

[28] Haifeng Xu, Kai Wang, Phebe Vayanos, and Milind Tambe. 2018. Strategic coordination of human patrollers and mobile sensors with signaling for security games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[29] Zhengyu Yin, Albert Xin Jiang, Milind Tambe, Christopher Kiekintveld, Kevin Leyton-Brown, Tuomas Sandholm, and John P Sullivan. 2012. TRUSTS: Scheduling randomized patrols for fare inspection in transit systems using game theory. *AI magazine* 33, 4 (2012), 59–59.