Abstraction in Opportunistic Crime Security Games: Extended Study

Chao Zhang, Victor Bucarey*, Ayan Mukhopadhyay†, Arunesh Sinha, Yundi Qian, Yevgeniy Vorobeychik†, Milind Tambe
University of Southern California, Los Angeles, CA, 90089, USA
*Universidad de Chile,Santiago, Region Metropolitana, Chile
†Vanderbilt University, Nashville, TN 37235, USA
{zhan661, aruneshs, yundi.qian, tambe}@usc.edu, *vbucarey@ing.uchile.cl,
†{ayanmukg, eug.vorobey}@gmail.com

ABSTRACT

In this paper, we aim to deter urban crime by recommending optimal police patrol strategies against *opportunistic criminals* in large scale urban problems. While previous work has tried to learn criminals' behavior from real world data and generate patrol strategies against opportunistic crimes, it cannot scale up to large-scale urban problems. Our first contribution is a game abstraction framework that can handle opportunistic crimes in large-scale urban areas. In this game abstraction framework, we model the interaction between officers and opportunistic criminals as a game with discrete targets. By merging similar targets, we obtain an abstract game with fewer total targets. We use real world data to learn and plan against opportunistic criminals in this abstract game, and then propagate the results of this abstract game back to the original game. Our second contribution is the layer-generating algorithm used to merge targets as described in the framework above. This algorithm applies a mixed integer linear program (MILP) to merge similar and geographically neighboring targets in the large scale problem. As our third contribution, we propose a planning algorithm that recommends a mixed strategy against opportunistic criminals. Finally, our fourth contribution is a heuristic propagation model to handle the problem of limited data we occasionally encounter in large-scale problems. As part of our collaboration with local police departments, we apply our model in two large scale urban problems: a university campus and a city. Our approach provides high prediction accuracy in the real datasets; furthermore, we project significant crime rate reduction using our planning strategy compared to current police strategy.

Categories and Subject Descriptors

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

General Terms

Security, Human Factors

Keywords

Security Games; Abstraction; Machine Learning

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

1. INTRODUCTION

Managing urban crime has always posed a significant challenge for modern society. Distinct from elaborately planned terrorists attacks, urban crimes are usually committed by *opportunistic criminals* who are less careful in planning the attack and more flexible in executing such plans [24]. Almost universally, preventive police patrolling is used with the goal of deterring these crimes. At the same time, opportunistic criminals observe the police deployment and react opportunistically . Therefore, it is very important to deploy the police resources strategically against informed criminals.

Previous work has tackled the problem of allocating police resources against opportunistic criminals. There are two approaches to recommend patrol strategies. The first approach is security games, such as Stackelberg Security Games [25] and Opportunistic Security Games [26], where the interaction between police officers and opportunistic criminals is modeled as a leader-follower game. Security games contain various extensions to handle different real world scenarios, but the models of adversary behavior are based on expert hypotheses, and lack detail as they are not learned from real-world data for defender's strategy and adversary's reaction. The second approach uses larger amounts of data, such as the patrol allocation history and corresponding crime report, to learn a richer Dynamic Bayesian Network (DBN) model [27] of the interaction between the police officers and opportunistic criminals. The optimal patrol strategy is generated using the learned parameters of the DBN. While this approach predicts criminals' behavior with high accuracy for the problem in which the number of target areas is small, it has three shortcomings: i) it cannot scale up to problems with a large number of targets; ii) the algorithm performs poorly in situations where the defender's patrol data is limited; iii) the planning algorithm only searches for a pure patrol strategy, which quickly converges to a predictable pattern that can be easily exploited by criminals.

In this paper, we focus on the problem of generating effective patrol strategies against opportunistic criminals in large scale urban settings In order to utilize the superior performance of DBN as compared to other models given ample data, we propose a novel abstraction framework. This abstraction framework is our *first contribution*. In this framework we merge the targets with similar properties and extract a problem with a small number of targets. We call this new problem the *abstract layer* and the original problem the *original layer*. We first learn in the abstract layer using the DBN approach [27] and generate the optimal patrol strategy, then we propagate the learned parameters to the original layer and use the resource allocation in the abstract layer to generate a detailed strategy in the original layer. By solving the problem hierarchically through multiple abstractions, we can generate the optimal strategy for the original scenario.

Our *second contribution* is a layer generating algorithm, for which (i) we model it as a districting problem and propose a MILP in order to merge targets in the original problem into geographically compact and contiguous *aggregated targets* keeping the similarity (defined later) within them as homogeneous as possible; (ii) we develop a heuristic to solve this problem in large scale instances; (iii) we propose two approaches to find the optimal aggregated targets. Our *third contribution* is a planning algorithm that generates an optimal mixed strategy against opportunistic criminals. We consider a mixed strategy because (i) it broadens the scope of the defender's strategies; (ii) previous pure strategies depended on the model getting updated periodically; as mentioned earlier, the model usually converged to a single pure strategy that is easy to exploit.

When the defender's patrol data is limited or even missing in the original layer, the learning approach in [27] overfits the data. Therefore, in order to solve this problem, we propose our *fourth contribution* which is a heuristic model to propagate important features from the abstract layer to the original layer. We use models from behavioral game theory, such as Quantal Response, to extract these features. In particular, we first approximate the learned DBN parameters in the abstract layer using behavioral parameters. Then the behavioral parameters are propagated to the original layer.

Finally, we evaluate our abstract game in two scenarios: the University of Southern California (USC) campus [27] and Nashville, TN. We obtain data in USC from [27]. Data in Nashville, TN is obtained as part of the collaboration with the local police department.

2. RELATED WORK

There are five threads of research that are related to our problem. The first line of work we compare with is game theoretic models, such as Stackelberg Security Games (SSG) [25], Opportunistic Security Games (OSG) [26], Patrolling security games (PSG) [5] and Pursuit Evasion Games (PEG) [16]. The interaction between police and criminals is modeled as a Security Game. While SSG is successfully applied in security domains to generate randomized patrol strategies., e.g., in counter-terrorism and fare evasion checks on trains [17], it *assumes* that attackers are perfectly rational. A lot of recent research has focused on attackers with bounded rationality. An example of such work is Opportunistic Security Games (OSG) [26]. In OSGs, attackers are opportunistic criminals who are boundedly rational in planning the attacks but more flexible in executing the plan. An optimal patrol strategy against such opportunistic adversaries is generated in OSGs. Recent work in leader-follower games, PSG, also has made progress in generating patrol strategies against adversaries in arbitrary topology [4]. Different types of adversaries in this game are considered in [6] while different security resources are considered in [3]. Another example of a game theoretic model is PEG, which models a pursuer attempting to catch an evader [16]. In PEG, the evader is trying to avoid capture and the pursuer is trying to capture the evader. However, as stated before, the adversary models in these games are hypothesized based on expert input and not detailed models — detailed in locations and time — learned from large amounts of real-world data that leads to the scale-up challenges addressed in our work.

The second area of work we compare with is data mining and machine learning in the criminology domain. Recent research uses real world crime data to analyze criminal behavior and recommend patrol strategies for police. In [9], the author summarizes the general framework in this domain. In [20], crime detection and crime pattern clustering is achieved through data mining; in [11], machine learning is used for criminal career analysis. However, in this area of research, only crime data is considered. It does not explicitly model and learn the interaction between police and these criminals from real world data; and nor does this work focus on planning police mixed strategies.

The third area of work we compare with is machine learning in game theory. In [27], the interaction between a criminal and the defender is modeled as a Dynamic Bayesian Network (DBN). Crime and patrol data are used to learn such interaction in this DBN and the defender's optimal strategy is generated. Unfortunately, this approach only works in small scale problems. When the number of targets increases, the time complexity and the number of unknown variables increase dramatically; we show in Section 6, that this approach fails to run when the number of targets increases beyond 20. In [7], the payoffs of attackers in SSGs are learned from their responses against the defender's strategy. However, in this approach, the goal is to show defender strategies that enable fast learning of the adversary's payoff instead of learning these models from existing detailed data of defender-adversary interactions. Another example of such work is Green Security Games (GSG)[12, 21], where poaching data may be used to learn a model of poachers' boundedly rational decision making; as noted in our earlier paper[27], our work complements theirs, and applying abstraction hierarchies introduced in this paper in GSGs remains an interesting issue for future work.

The last thread of recent research we compare with is the abstract game that is widely used in large incomplete information games such as Texas Hold'em [13, 15]. There are a number of different approaches including both lossless abstractions [14] and lossy abstractions [22]. In [10] and [1], sub-games are generated to calculate the Nash equilibrium in a normal form games. Abstractions have also been brought into security games. In [2], abstraction is used to design scalable algorithms in PSGs. However, these works focus on clustering similar actions, strategies or states to formulate a simpler game. In our situation, we are physically merging the similar targets to generate simpler games. The criteria of merging targets is different from that of merging actions, strategies or states. Our differing criteria and approach for merging targets, different means of propagating results of our abstractions, and our learning from real-world crime data set our work apart from this work.

3. PROBLEM STATEMENT

TARGET	Case	Type	Date	Time
T2	C1	Theft	4/3/2012	8:00
T9	C2	Traffic	4/3/2012	9:30
T3	C3	Property	4/3/2012	12:00
T5	C4	Theft	4/3/2012	13:13
T6	C5	Service	4/3/2012	15:32
T1	C6	Domestic	4/3/2012	17:00

Figure 1: Sample Crime Report

TARGET	Officer Info	
T1	P1	Mike
T2	P2	Peter
T3	P3	William
T4	P4	Ben
T5	P5	Jack
T6	P6	Lucy

Figure 2: Sample Schedule

In this paper, we focus on limiting opportunistic crimes in large scale urban areas. Such large scale areas are usually divided into N targets by the defenders. At the same time, defenders divide the time into patrol shifts. T denotes the total number of shifts. At the beginning of each patrol shift, the defender assigns each available patrol officer to a target and the officer patrols this target in this shift. The criminals observe the defender's allocation and seek crime opportunities by deciding the target to visit. In order to learn the criminal's opportunistic reaction to the defender's allocation, two categories of data are required for T shifts. The first is about crime activity which contain crime details. Figure 1 shows a snapshot of this kind of data in a campus region. In this paper, we only consider the time and location information of crimes, ignoring the difference among different types of crimes. Therefore, we can summarize the crime report into a table like Table 1. In this table, columns represents the index of each target while rows represents total number of shifts, 1...T. Each element in the table represents the number of crimes at the corresponding target in that shift. $N \times T$ data points are recorded in the table.

Shift	1	2	3	...	N
1	1	1	2	...	2
2	1	1	1	...	1
3	2	1	1	...	1

Table 1: Crime data

The second category is the patrol allocation schedule at these shifts. The snapshot of such data is shown in Figure 2. We ignore the individual difference between officers and assume that the officers are homogeneous and have the same effect on criminals' behavior. Therefore, only the number of officers at each target and shift affects criminals' behavior and we can summarize the patrol data in the similar manner as crime reports, which is shown in Table 2.

Shift	1	2	3	...	N
1	2	1	1	...	1
2	1	1	2	...	2
3	2	1	1	...	1

Table 2: Patrol data

Given the available data for crime and patrol officers, our goal is to recommend efficient patrol strategies to prevent opportunistic crimes in problems with a large number of targets. To begin with, we learn criminals' behavior from data and apply the abstract game framework to hierarchically learn the criminal's behavior. Next, we propose a planning algorithm that generates the mixed strategy that optimizes the defender's utility against the learned behavior of criminals.

4. ABSTRACT GAME

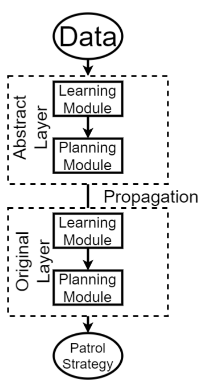

Figure 3: Game Abstraction

Even though previous approaches [27] deal with opportunistic crimes, they cannot directly be applied to large scale problems. There are two reasons. First, over-fitting is inevitable in the learning process of large scale problems. The number of unknown variables in the learning process is $O(N^2)$ while the number of data points is $O(N \times T)$ [27]. When N increases, the number of variables gets close to the number of data points and causes overfitting. The second reason is the runtime. The complexity of previous approaches is at least $O(N^{C+1}T)$ where C is the largest value that any variable in the model of [27] can take and it grows quickly with N. In fact, our experiments shows that the algorithm does not converge in one day even with $N = 25$. Therefore, we propose the abstract game framework to deal with opportunistic crimes in large scale urban areas.

The idea of abstracting the most essential properties of a complex real problem to form a simple approximate problem has been widely used in the poker game domain [13]. Using such an abstraction the problem can be solved hierarchically and a useful approximation of an optimal strategy for the real problem is provided. In this paper, we use the concept of abstraction to transform the large scale urban area problem into a smaller abstract problem and solve it hierarchically. Figure 3 illustrates the four steps in our game abstraction framework. First, we need to generate the *abstract layer* from the *original layer* (Section 4.1). Targets that have similar properties are merged together into *aggregated targets*. The set of aggregated targets is called the *abstract layer* while the set of original targets is called the *original layer*. Currently we only consider two layers: the original layer and the abstract layer. If the problem in the abstract layer is still too large to solve, we need to do further abstraction, which we will discuss in Section 4.5. After we obtain the abstract layer, the second step is to learn the criminal's behavior and generate an optimal patrol strategy in the abstract layer (Section 4.2). The third step is to propagate information, such as criminal behavior features, from the abstract layer to the original layer (Section 4.3). Finally, we use the information from the abstract layer and data in the original layer to learn the criminal's behavior and generate an optimal patrol strategy in the original layer (Section 4.4).

4.1 Layer Generating Algorithm

We model the layer generation as a Districting Problem [18, 8]. The districting problem is the well known problem of dividing a geographical region into balanced subregions with the notion of balance differing for different applications. For example, police districting problems focus on workload equality [8]. Our layer generation is a districting problem that group targets in the original layer into *aggregated targets* in the abstract layer. However, distinct from the classic Districting problem where the resources are balanced among different *aggregated targets*, in our problem, we try to maximize the similarity of the targets inside the same *aggregated target*. We do so by modeling the similarity of targets within each aggregated target and use this similarity measure as one of the criteria in the optimization formulation of our problem.

Let $I = \{1, \ldots, N\}$ be a set of targets in the original layer. A partition of size K of this set I is a collection of sets $\{I_k\}_{k=1}^K$ such that $I_k \neq \emptyset$ for all $k \in \{1, \ldots, K\}$, $I_k \cap I_l = \emptyset$ for all $k, l \in \{1, \ldots, K\}, k \neq l$ and $\bigcup_{k=1}^K I_k = I$. $\{I_k\}_{k=1}^K$ is the set of the aggregated targets in the abstract layer. Let $\mathcal{P}_K(I)$ denote the set of all partitions of I of size K. Given $I_k \subset I$ we define its inner Dissimilarity as $Dis(I_k) = \sum_{i,j \in I_k} Dis_{ij} = \sum_{i,j \in I_k} |c_1^i - c_1^j| + |c_0^i - c_0^j|$. Also we define its Inertia as $In(I_k) = \min_j \sum_{i \in I_k} d_{ij}$, with d_{ij} denoting the physical distance between the geometric centers of targets i, j. In our districting process we want to find a partition which achieves both low inner Dissimilarity and Inertia over all elements of the partition. Given $\alpha > 0$ as a normalization parameter, we define the information loss function $L_I(K)$ as the lowest cost with a partition of size K, mathematically $L_I(K) = \min_{\{I_k\}_{k=1}^K \in \mathcal{P}_K(I)} \sum_{k=1}^K \alpha In(I_k) + Dis(I_k)$.

LEMMA 1. *The information loss decreases with K, that is $L_I(K+1) \leq L_I(K)$.*

The proof of Lemma 1 is in the appendix (`http://bit.ly/1ND8liH`) . Based on these three principles, we propose a mixed integer linear program (MILP) to solve the districting problem. We apply an extension of the capacitated K-median problem with $K = n$. While the capacitated K-median problem [23] satisfies the scalability constraint by setting a maximum capacity for each aggregated target, it cannot handle the geometric constraints such as contiguity. A counterexample is shown in the appendix. In this paper, we handle the geometric constraints by considering the inertia of each aggregated target as part of the information loss function.

$$
\begin{aligned}
\min_{x,y,z} \quad & \alpha \sum_{i,j} d_{ij} y_{ij} + \sum_{ik} z_{ik} & \\
s.t. \quad & \sum_j y_{ij} = 1 & \forall i \in I \\
& y_{ij} \leq x_j & \forall i, j \in I \\
& \sum_j x_j = n & \\
& \sum_j y_{ij} \leq n & \forall j \in I \\
& z_{ik} \geq Dis_{ik}(y_{ij} + y_{kj} - 1) & \forall i, k, j \in I \\
& z_{ik} \geq 0 & \forall i, k \in I \\
& y_{ij} + y_{kj} \leq 1 & \forall j \in I \\
& y_{ij}, x_j \in \{0, 1\} & \forall i, j \in I
\end{aligned} \tag{1}
$$

x_j is a binary variable. It is 1 if the target j is the center of an aggregated target and 0 otherwise. The variable y_{ij} takes the value 1 when the target i is allocated to the aggregated target centered in j and 0 otherwise. The variable z_{ik} is a continuous non-negative variable that takes the value Dis_{ik} when target i and target k are allocated to the same aggregated target, otherwise z_{ik} is 0. The objective function is the weighted sum of inertia and dissimilarity. α represents the trade-off between geometric shape and the similarity within each aggregated target.

The first set of constraints ensures that every target is allocated to an aggregated target. The second set of constraints ensures that the center of an aggregated target belongs to this aggregated target. The third expression states that there are n aggregated targets. The fourth set of inequalities ensures the size of every aggregated target to be no greater than n. The fifth and sixth constraint ensures that z_{ik} will take the value Dis_{ik} when target i and target k are allocated to the same aggregated target, otherwise z_{ik} will be 0. The seventh constraint is an example of environmental constraints that target i and target k cannot be in the same aggregated target.

Directly solving this MILP is NP-hard [19]. Therefore we use the heuristic constraint generation algorithm to approximately solve the problem. The detail of this algorithm is in the appendix.

4.2 Abstract Layer

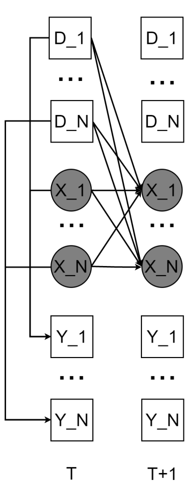

Figure 4: DBN framework

Learning Algorithms: As noted earlier, having generated the abstract layer, the next step is learn the adversary model at the abstract layer. As stated before, the Dynamic Bayes Network (DBN) learning algorithm presented in [27] could not be used in the original layer due to scaling difficulties; however, with a sufficiently small number of targets in the abstract layer, we can now use it. To illustrate its operation, we reproduce the operation with N targets as shown in Figure 4. Three types of variables are considered in the DBN: squares in the top represent the number of defenders at aggregated target i during shift t, $D_{i,t}$, squares in the bottom represent the number of crimes at aggregated target i during shift t, $Y_{i,t}$, while circles represents the number of criminals at aggregated target i during shift t, $X_{i,t}$. As shown in Figure 4, there are two transitions in the DBN: the criminal's transition from shift t to $t+1$, which is modeled as the transition probability and the crime transition at shift t, which is modeled as the crime output probability. Mathematically, a transition probability is defined as $P(X_{i,t+1}|D_{1,t}, ..., D_{N,t}, X_{1,t}, ..., X_{N,t})$ and the crime output probability is defined as $P(Y_{i,t}|D_{1,t}, ..., D_{N,t}, X_{1,t}, ..., X_{N,t})$. This model uses two matrices to represent the transition probabilities, the movement matrix A which consists of all the criminal's transition probability $P(X_{i,t+1}|D_{1,t}, ..., D_{N,t}, X_{1,t}, ..., X_{N,t})$ and the crime matrix B which consists of all the crime output probability $P(Y_{i,t}|D_{1,t}, ..., D_{N,t}, X_{1,t}, ..., X_{N,t})$. A and B contains $C^N \times C^N \times C^N$ unknown parameters.

Given available data about $D_{i,t}$ (patrol schedule), $Y_{i,t}$ (crime report), this model applies the Expectation Maximization algorithm to learn A and B while estimating $X_{i,t}$. The detail of this learning model is present in [27]. The novelty in this paper is propagating adversary behavior parameters (A and B) from the abstract layer to the original layer, which we discuss in Section 4.3; but we do that, we discuss planning in the abstract layer.

Planning Algorithms: In this paper, we focus on planning with mixed strategies for the defender rather than the pure strategy plans from previous work [27]. This change in focus is based on two key reasons. First, this change essentially broadens the scope of the defender's strategies; if pure strategies are superior our new algorithm will settle on those (but it tends to result in mixed strategies). Second, previous work [27] on planning with pure strategies depended on repeatedly cycling through the following steps: planning multiple shifts of police allocation for a finite horizon, followed by updating of the model with data. This approach critically depended on the model getting updated periodically in deployment. Such periodic updating was not always easy to ensure. Thus, within any one cycle, the algorithm in [27] led to a single pure strategy (single police allocation) being repeated over the finite horizon in real-world tests as it tried to act based on the model learned from past data; such repetition was due to a lack of updating of the criminal model with data, and in the real-world, the criminals would be able to exploit such repetition. Instead, here we plan for a mixed strategy. We assume that the model updates may not occur frequently and as a result we plan for a steady state.

We model the planning procedure as an optimization problem where our objective is to maximize the defender's utility per shift. After the defenders' (mixed) strategy is deployed for a long time, criminals receive perfect information of the strategy and their (probabilistic) reaction will not change over time. As a result, the criminals' distribution becomes stationary and this is called criminals' stationary state. In our case, ergodicity guarantees unique stationary state (see appendix). Our planning algorithm assumes criminals' stationary state when maximizing the defender's utility. We define defender's utility as the negation of the number of crimes. Therefore, the objective is to minimize the number of crimes that happen per shift in the stationary state. Let's define $I = \{i\}$ as the set of aggregated targets, D as the total number of defenders that are available for allocation; $d_I = \{d_i\}$ as the set of defender's allocation at target set I, $x_I = \{x_i\}$ as the set of criminal's stationary distribution at target set I with respect to defender's strategy d_I and $y_I = \{y_i\}$ as the set of expected number of crimes at target I. Note that C is the largest value that the variables D_i, X_i and Y_i can take. The optimization problem can be formed as follows:

$$
\begin{aligned}
\underset{d_I}{\text{minimize}} \quad & \textstyle\sum_{i \in I} y_i \\
\text{subject to} \quad & 0 \leq x_i \leq C,\ i \in I, \\
& 0 \leq d_i \leq C,\ i \in I, \\
& \textstyle\sum_{i \in I} d_i \leq D, \\
& y_i = \textstyle\sum_{Y_{i,t}} Y_{i,t} \cdot \\
& P(Y_{i,t}|d_1, ..., d_N, x_1, ..., x_N),\ i \in I, \\
& x_i = \textstyle\sum_{X_{i,t+1}} X_{i,t+1} \cdot \\
& P(X_{i,t+1}|d_1, ..., d_N, x_1, ..., x_N),\ i \in I.
\end{aligned} \tag{2}
$$

In this optimization problem, we are trying to minimize the total number of crimes occurring in one shift while satisfying five sets of constraints. The first two constraints ensure the defender and criminal's distribution are non-negative and no more than an upper bound C. The third constraint represents the constraint that the number of deployed defender resources cannot be more than the available defender resources. The fourth constraint is the crime constraint. It sets y_i to be the expected number of crime at target i. The last constraint is the stationary constraint, which means that the criminals' distribution is not changing from shift to shift with respect to the patrol strategy d_I. The transitions are calculated by movement matrix A and crime matrix B. The details of the crime and stationary constraint are shown in the appendix.

4.3 Propagation of learned criminal model

In the previous section, we generate the patrol allocation for the aggregated targets in the abstract layer. In order to provide patrolling instructions for the original layer, we propagate the learned criminal model in the abstract layer to the original layer. We need to address two cases: when there is no detailed patrol data and when there is. In particular, we have found that some police departments record the location of police patrols in detail at the level of targets in the original layer, but many others specifically only keep approximate information and do not record details (even if they record all crime locations in detail); thus leading to the two cases. We start by describing the case with sufficient patrol data in the original layer.

Direct learning (sufficient data): When there is detailed patrol data in the original layer and nothing is approximated away, we know the numbers of police at each target in the original layer at each shift. Then, we can directly learn A and B in this DBN. The learning algorithm is same as that applied in the abstract layer. The data used in the algorithm is the crime report and patrol schedule inside each *aggregated target*. While we directly learn A and B, computation of the patrol strategy at the abstract layer affects the patrol strategy in the detailed layer as discussed in Section 4.4.

Parameter Propagation (limited data): If the patrol data in the original layer is limited, the DBN model that we learned in the original layer will be inaccurate if we still apply the same learning algorithm as the abstract layer to learn matrices A and B. One remedy is to provide addition criminal information to the original layer from the abstract layer to help the process of learning criminal model in the original layer. However, in the abstract layer, movement matrix A and crime matrix B represent the criminal's behavior in aggregated targets. It cannot directly describe the criminal's behavior in the targets in the original layer. Therefore, we propose a human behavior based model of extracting behavior parameters from A and B in the abstract layer. Then we set these behavior parameters of an aggregated target as the behavior parameters for the targets contained within this aggregated target in the original layer.

Parameter extraction: We introduce the process of using a human behavior model to extract the behavior parameters from A and B. The basic assumption of a human behavior model is that the criminal follows certain patterns when moving from shift to shift. Specifically, the criminals follow the movement by the well established Quantal Response (QR). In the learning algorithm [27], one simplification made was breaking down the criminals' transition probabilities into marginal probability $P(X_{j,t+1}|D_{i,t}, X_{i,t})$ which represents the movement of a criminal from target i to target j. Based on the Quantal Response model, we approximate this movement using the following equation: $P(X_{j,t+1} = 1) = \frac{e^{Att_j}}{\sum_{n \in N} e^{Att_n}}$ where Att_n is the attractiveness property of target n. In the DBN, the movement depends not only on the attractiveness, but also on the allocation of defenders and criminals at previous shift. Therefore, we formulate $\hat{P}(X_{j,t+1} = 1|D_{i,t}, X_{i,t})$ as ($\lambda_i, \mu_i \geq 0$):

$$
\begin{cases}
\frac{e^{Att_j}}{\sum_{n \in N} e^{Att_n}} \cdot e^{\lambda_i X_{i,t} + \mu_i D_{i,t}}, & \text{if } i \neq j \\
\frac{e^{Att_j}}{\sum_{n \in N} e^{Att_n}} \cdot e^{\lambda_i X_{i,t} - \mu_i D_{i,t}}, & \text{otherwise}
\end{cases} \tag{3}
$$

The reason for the above effect of defender is that the defender at target i disperses criminals to other targets. However, λ, μ and Att are not known and we need to learn them from data. Our approach to compute λ, μ and Att is to find their values that minimize the L_1 distance between $\hat{P}(X_{j,t+1} = 1|D_{i,t}, X_{i,t})$ and the learned marginal probability $P(X_{j,t+1} = 1|D_{i,t}, X_{i,t})$. We can formulate this problem as the following optimization:

$$
\begin{aligned}
\min_{Att,\lambda,\mu} \textstyle\sum_{i,j,D_{i,t},X_{i,t}} \quad & ||P(X_{j,t+1} = 1|D_{i,t}, X_{i,t}) - \\
& \hat{P}(X_{j,t+1} = 1|D_{i,t}, X_{i,t})|| \\
\text{subject to} \quad & \mu_i \geq 0, \lambda_i \geq 0, i = 1, ..., N
\end{aligned}
$$

The constraints represent the positive effect of number of criminals on the transition probability and more defenders lead to faster dispersion of criminals. λ, μ and Att are the behavior parameters that we propagate to original layer.

Since λ and μ represent the influence of the number of criminals and number of defenders on the criminals' movement in the aggregated target, it is reasonable to assume that the criminals' movement in the targets that belong to the aggregated target inherit these parameters. In other words, this means that the influence of the number of criminals and defenders is the same within the aggregated target. At the same time, Att measures the availability of the crime opportunities. Therefore, within one aggregated target, the attractiveness is distributed among the targets proportional to the total number of crimes in each target. For example, if the attractiveness of an aggregated target I (made up of I_1 and I_2) is 0.6, the total number of crimes at target I_1 is 80 while that at target I_2 is 40, then the attractiveness of A_1 is 0.4 while that of A_2 is 0.2. λ, μ and Att for each target are the behavior parameters that will be used in crime and stationary constraints in the planning algorithm.

4.4 Computing Strategy in the Original Layer

In the previous section, we generated the adversary behavior parameters in the original layer. In order to provide patrolling instructions for the original layer, we utilize the strategy in the abstract layer to assign resources in the original layer. Then, combined with the propagated adversary behavior parameters we generate the strategy at the original layer.

Resource Allocation: In the abstract layer, the optimal strategy recommends the number of resources allocated to each *aggregated target*. We use this recommendation as a constraint on the number of resources in planning within the *aggregated targets* at the original layer. For example, the abstract layer may provide 0.8 as the allocation to an aggregated target say X; then we plan patrols in X in the original layer using 0.8 as the total number of resources.

Next, in the original layer, we treat each *aggregated target* in the abstract layer as an independent DBN as shown in Figure 4. The same algorithm for generating a mixed strategy in the abstract layer can be applied in each of the independent DBNs. The optimization problem is the same as Equation 2. D is the total number of resources allocated to these aggregated targets (e.g., 0.8 to target X).

In addition, the formulations of crime and stationary constraints required in the computation of the mixed strategy are different for the scenario with sufficient and limited data. For the scenario with sufficient data these constraints are formulated using the parameters A and B of the DBN that is learned in this original layer. For the scenario with limited data the propagated values of λ, μ and Att are used to estimate the the A and B parameters for the DBN representation of the adversary behavior in the original layer. The estimation is the inversion of parameter extraction, and it happens in the original layer. For example, we use Equation 3 to estimate the parameters using λ, μ and Att. The details are presented in the appendix. Then, these reconstructed A and B are used to formulate the crime and stationary constraints.

4.5 Extended Abstract Game

When $n^2 < N$, we can use two layers of abstraction to solve the problem. However, when the real problem has $N > n^2$ targets, even two layered abstraction does not suffice since there must be a layer in the game with more than n targets. Therefore, we propose the multiple layer framework to handle problems with an arbitrarily large number of targets. This framework is an extension of the two layer abstract game. We apply an iterative four step process. As a first step, we need to decide the number of layers as well as the districting of targets for each of the layers. Considering the scalability constraints (recall that there cannot be more than n targets within each aggregated target), the number of layers is $M = \lfloor \log_n N \rfloor + 1$. We denote the original layer as Layer 1 and the layer directly generated from Layer m as layer $m + 1$. In this notation, the topmost abstract layer is Layer M. The second step is learning criminals behavior in the top layer. The third step is to generate a patrol strategy at this layer. The fourth step is to propagate parameters to the next layer. We keep executing steps two to four for each layer until we reach the original layer. At each layer, we decide whether to do parameter propagation based on the availability of the patrol data. If we have sufficient patrol data at layer m, we do direct learning at layer m. Otherwise, we do parameter propagation from layer $m + 1$ to layer m.

We propose three different layer generation algorithms. The first algorithm is the direct algorithm. For example, if $N = 50$ and $n = 5$. Then, there should be $M = 3$ layers. For layer 1, there will be 50 targets. For layer 2, the number of targets could be any integer between 10 to 25. For layer 3, the number of targets can be 2 to 5. The direct learning tries all the combinations of three layers and runs the MILP for each combination to generate the optimal segmentation. It calls the MILP in Section 4.1 for $O(N^M \cdot M)$ times; the second algorithm is a dynamic programming approach that ensures the solution is globally optimal. The MILP is called $O(N^2 \cdot M)$ times; the third algorithm is the greedy algorithm that sets the number of targets to be maximum, which for the mth layer is n^{M+1-m}. The number of calls is M while the solution is not necessarily optimal. Details are in the appendix.

5. REAL WORLD VALIDATION

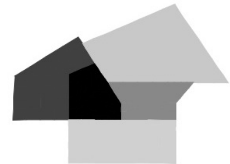

Figure 5: Campus map 1

Figure 6: Campus map 2

We use two sets of real world data to validate the game abstraction framework. In the first case we use the data from the University of Southern California (USC) campus that is provided by [27]. We thank the authors for providing three years (2012-2014) of crime report and patrol schedule from the USC campus. The number of total crime events is on the order of 10^2. [27] reports that the campus patrol area (USC campus and its surroundings) is divided into five patrol areas, which are shown in Fig 5. In order to make the patrols more efficient, the police officers wish to further divide the whole campus into 25 patrol areas and get patrol recommendations on these 25 patrol areas. There are two tasks for us, (a) starting from city blocks (there are 298 city blocks and they form the basis of the USC map), create 25 separate "targets", as in our layer generation problem; (b) generate an optimal patrol strategy for these 25 targets. The creation of 25 targets is also a districting problem and the technique in Section 4.1 can be directly applied. The 25 targets generated by the districting algorithm is shown in Figure 6. We treat these 25 targets as the original layer. n is set to be 5 as the runtime of learning and planning algorithm with $n = 5$ is reasonably small. So then we use two layer game abstraction to solve this problem with 25 targets. The abstract layer is the five patrol areas in Fig 5. This is because of the center area (the darkest area) is the campus itself and is separated from its environment by fences and gates. These environmental constraints cause our layer generation to automatically create the area into 5 targets as shown in Figure 5. Additionally, police only record their presence in the five areas, and thus, we do not have detailed police presence data; as a result, we use our behavior learning to propagate parameters from the abstract layer to the original layer.

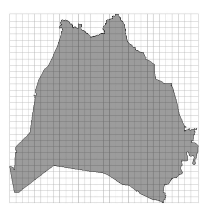

Figure 7: City map

In the second case, we use data about crime and detailed police patrol locations in Nashville, TN, USA. The data covers a total area of 526 sq. miles. Only burglaries (burglary/breaking and entering) have been considered for the analysis. Burglary is the chosen crime type as it is a major portion of all property crimes and is well distributed throughout the county. Data for 10 months in 2009 is used. The number of total crime events is on the order of 10^3. Observations that lacked coordinates were geocoded from their addresses. Police presence is calculated from GPS dispatches made by police patrol vehicles. Each dispatch consists of a unique vehicle identifier, a timestamp and the exact location of the vehicle at that point in time. We divide the whole city into $N = 900$ targets as shown in Figure 7. Since n is 5, the number of layers we need is $M = \lfloor \log_5 900 \rfloor + 1 = 5$. We use the multiple layer abstraction framework to solve this problem.

6. EXPERIMENTAL RESULTS

Experiment setup. We use MATLAB to solve our optimization problems. There are two threads of experiments, one on the USC campus problem and the other on Nashville, TN problem. To avoid leaking confidential information of police departments, all

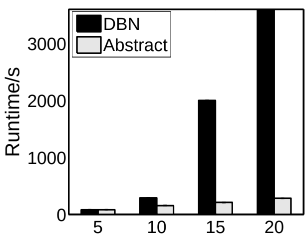

Figure 8: **Runtime**

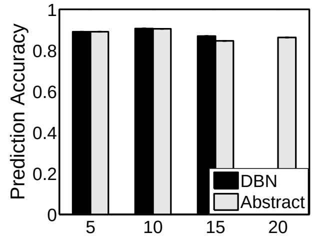

Figure 9: **Accuracy**

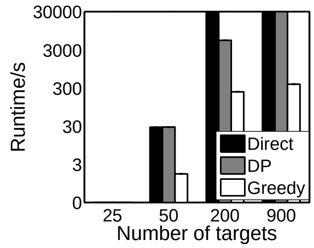

Figure 10: **Runtime**

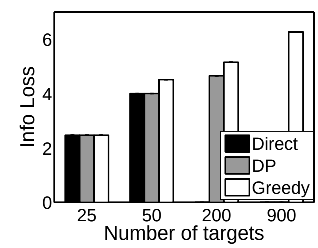

Figure 11: **Information Loss**

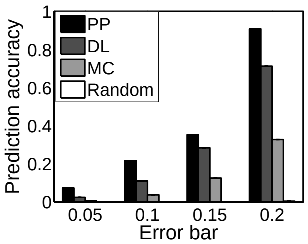

Figure 12: **Accuracy (USC)**

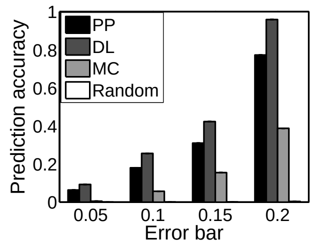

Figure 13: **Accuracy (city)**

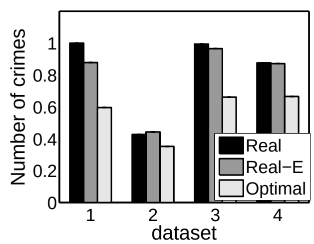

Figure 14: **Plan (USC)**

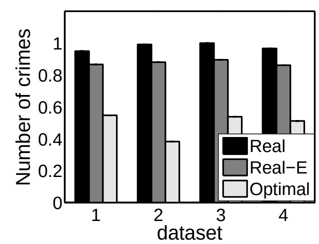

Figure 15: **Plan (city)**

crime numbers shown in the results are normalized. The experiments were run on a machine with 2.4 GHz and 16 GB RAM.

Game Abstraction Framework: Our first experiment is on comparing the performance of our game abstraction framework with the DBN framework proposed in [27] for large scale problems. Since the DBN framework cannot even scale to problems with 25 targets, in this experiment we run on problems with subsets containing N targets ($5 \leq N < 25$) out of these 25 targets in the USC campus. As shown in Figure 8, we compare the runtime of these two frameworks. The x-axis in Fig. 8 is the number of targets N in the problem. For each N, we try ten different subsets and the average runtime is reported. The y-axis indicates the runtime in seconds. The cut-off time is 3600s. As can be seen in Figure 8, the runtime of the DBN framework grows exponentially with the scale of the problem and cannot finish in an hour when $N = 20$. At the same time, the runtime of the game abstraction framework grows linearly with the scale of the problem. It takes less than 5 minutes to solve the problems with $N = 20$. This indicates that the DBN framework fails to scale up to large scale problems while the game abstraction framework can handle many more targets.

In Figure 9 we compare the *prediction accuracy* of these two different frameworks. We divide the 36 months' data sets into two parts, the first 35 months' data is used for learning while we predict the crime distribution for the last month and compare it with the real crime data in that month. For every target and every shift, we measure the *prediction accuracy* as probability that the difference of real crime distribution and our predicted crime distribution are within the errorbar. For example, for target i and shift t, our prediction is that there is 30% probability that no crime occurs and 70% that one crime occurs while in the data there is one crime at target i in shift t. Then, the prediction accuracy for target i for shift t is 0.7. The reported accuracy is the average accuracy over all targets and all shifts over all ten different subsets. The higher the accuracy, the better our prediction. As can be seen in Figure 9, the game abstraction framework achieves similar prediction accuracy compared to the DBN algorithms given any number of targets in the problem. This indicates that even through information may be lost during the abstraction, the game abstraction framework captures important features of the criminal and performs as well as the exact DBN framework while running 100 of times faster.

Layer Generation Algorithm: Next, we use the data from the city to evaluate the performance of our layer generation algorithms. Again, we run the layer generation algorithms on problems with subsets containing N targets ($N \leq 900$) out of the 900 targets in the city map. For each N, we try ten different subsets and report the average value except when $N = 900$ for which only one subset is possible. Figure 10 compares the runtime of different layer generation algorithms in log format. Three different algorithms are compared, the direct algorithm (Direct) that traverses all possible layer combinations; the dynamic programming algorithm (DP) and the greedy algorithm (Greedy). The x-axis in Fig. 10 is the number of targets N. For $N = 25$, two layers are needed; for $N = 50$, three layers are needed; for $N = 200$, four layers are needed and for $N = 900$, five layers are needed. The y-axis is the runtime of different algorithms in seconds. The cut-off time is set at 36000s. When $N = 25$, the runtime of these three algorithms are the same because the layer generation is unique. The number of targets in layer 2 is 5. When $N = 50$, the runtime of the direct algorithm is the same as that of the DP algorithm while the runtime of the greedy algorithm is significantly lower. When $N = 200$, the direct algorithm cannot finish in 10 hours; the DP algorithm takes around five hours while greedy algorithm finishes in less than 10 minutes. When $N = 900$, both direct learning and DP are cut off while the runtime for greedy is less than 15 minutes. This validates our theoretical result that the runtime of direct algorithm grows exponentially with the scale of the problem, that of DP grows polynomially and that of greedy algorithm grows linearly with the number of layers. Since both direct and DP algorithm cannot scale up to the problem with $N = 900$, we use the greedy algorithm as the layer generation algorithm in the city problem.

In Figure 11, we compare the information loss of different layer generation algorithms. The information loss is defined as the objective in Equation 1. As can be seen in Fig. 11, the information loss of DP is the same as that of direct learning in any situations. This is because DP ensures a globally optimal solution. At the same time, the information loss of the greedy algorithm is higher than that of the DP algorithm but no more than 15% higher. This indicates that while greedy algorithm cannot ensure global optimal information loss, it can reach a good approximation in reasonable runtime.

Learning: Third, we evaluate the performance of our learning algorithm. Game abstraction is used for both problems and we evaluate the predictions in the original layer. The result shown in

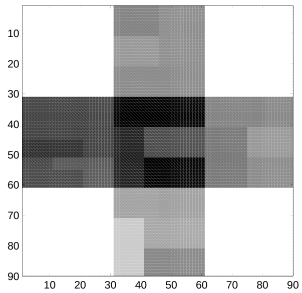

Figure 16: **HeatMap (No Defender)**

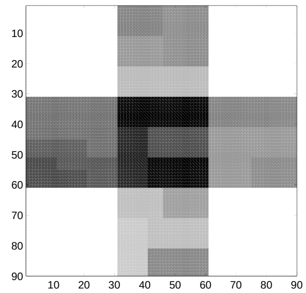

Figure 17: **HeatMap (Random)**

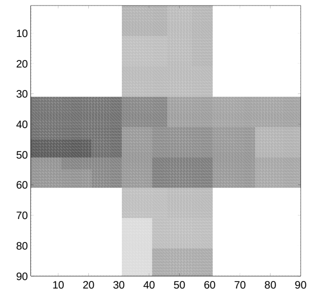

Figure 18: **HeatMap (Optimal)**

Figure 12 and Figure 13 compares the prediction accuracy of different algorithms in USC campus and the city problem respectively. Four different algorithms are compared: (1) the Random approach, in which the probabilities of each situation are the same (Random), (2) Markov Chain approach, in which the crimes are modeled as a Markov Chain without hidden state (MC), (3) game abstraction with direct learning for both the abstract and original layer (DL) and (4) game abstraction with parameter propagation in the original layer (PP). We divide the whole data sets into four equal parts. For each part, the first 90% of data is used for training while we test on the last 10% of data. The x-axis in Fig. 12 and 13 is the errorbar we use for calculating accuracy. The higher this errorbar is, the more error we are allowed in the prediction, y-axis indicates the prediction accuracy on the test set. As can be seen in both figures, the accuracy of both game abstraction based approaches are higher than that of the baseline MC and random algorithm in all the test sets. This indicates that game abstraction models help improve the prediction in large scale problems. In addition, parameter propagation at the original layer outperforms direct learning at this layer in the USC problem in Figure 12. Direct learning outperforms parameter propagation in Nashville problem in Figure 13. This is because the patrol data at the original layer in USC is limited. That is, only the aggregate number of police resources over several targets is available while the resources at each target remain unknown. Parameter propagation is better at handling limited patrol data. However, the patrol data is adequate in the city problem and direct learning is a better fit in such situations. Therefore, in the planning section, we use parameter propagation as the learning algorithm in the USC and direct learning as the learning algorithm in Nashville.

Planning: Next, we evaluate the performance of our planning algorithm in both the problems. Figure 14 and 15 compare strategies generated using the game abstraction framework with the actual deployed allocation strategy generated by the domain experts. Three different scenarios are compared: the real number of crimes, shown as Real; the expected number of crimes with manually generated strategies and learned adversary model with game abstraction, shown as Real-E and the expected number of crimes with the optimal strategy computed using game abstraction, shown as Optimal. As shown in Figure 14 and 15, the expected number of crime with manually generated strategy is close to the real number of crimes, which indicates game abstraction model captures the feature of criminals and provide good estimation of the real number of crimes. In addition, strategy generated using the game abstraction is projected to outperform the manually generated strategy significantly. This shows the effectiveness of our proposed patrol strategy as compared to the current patrol strategy.

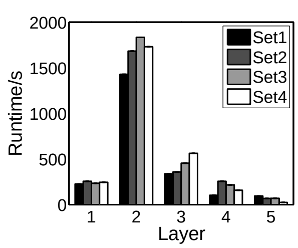

Figure 19: **Runtime**

Figure 16, 17 and 18 shows the crime heat map with different patrol strategies in USC campus problem. The 'heat' represents the crime probability of criminals at each target. The darker this color is, the higher the crime probability is. In Figure 16, there is no defender at any targets; in Figure 17, 5 defenders are randomly protecting the 25 targets on campus while in Figure 18, 5 defenders are allocated following the recommendation from our system. As can be seen in these three figures, the total crime probability is the highest when there is no defender. At the same time, even thought the number of resources are the same, the crime probability with optimal strategy is significantly lower than that with random strategy. This indicates that: i) the patrol officers can help reduce the crime rate in urban areas; ii) our planning algorithm can significantly reduce the crime rate by modeling the criminal's behavior.

Runtime: Finally, we break down the total runtime of the game abstraction framework in the city problem layer by layer and show it in Figure 19. The x-axis is the index of the layer, which goes from the original layer (Layer 1) to the top layer (Layer 5). The y-axis is the total runtime of the propagation, learning and planning algorithm in that layer. As can be seen, the runtime increases as the layer index decreases except for Layer 1. This is because in greedy layer generation, for the fifth layer the number of targets is 5, and for the fourth layer it is 5^2, for third layer it is 5^3, for the second layer it is 5^4 but for the first layer it is only 900. Therefore, the number of targets within each *aggregated target* in layer two is less than $3 < n = 5$. Therefore, the runtime in layer 1 is faster. However, the total runtime of the whole process is less than an hour in each data set. Therefore, the game abstraction framework can be extended to large scale problems with reasonable runtime performance.

7. CONCLUSIONS

This paper introduces a novel game abstraction framework to learn and plan against opportunistic criminals in large-scale urban areas. First, we model the layer-generating process as a districting problem and propose a MILP based technique to solve the problem. Next, we propose a planning algorithm that outputs randomized strategies. Finally, we use a heuristic propagation model to handle the problem with limited data. Experiments with real data in two urban settings shows that our framework can handle large scale urban problems that previous state-of-the-art techniques fail to scale up to.

REFERENCES

[1] A. Basak and C. Kiekintveld. Abstraction using analysis of subgames. In *IJCAI Workshop on Algorithmic Game Theory*, 2015.

[2] N. Basilico and N. Gatti. Automated abstractions for patrolling security games. In *AAAI*, 2011.

[3] N. Basilico and N. Gatti. Strategic guard placement for optimal response toalarms in security games. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1481–1482. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[4] N. Basilico, N. Gatti, and F. Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 57–64. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

[5] N. Basilico, N. Gatti, and F. Amigoni. Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder. *Artificial Intelligence*, 184:78–123, 2012.

[6] N. Basilico, N. Gatti, T. Rossi, S. Ceppi, and F. Amigoni. Extending algorithms for mobile robot patrolling in the presence of adversaries to more realistic settings. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 02*, pages 557–564. IEEE Computer Society, 2009.

[7] A. Blum, N. Haghtalab, and A. D. Procaccia. Learning optimal commitment to overcome insecurity. In *Advances in Neural Information Processing Systems*, pages 1826–1834, 2014.

[8] V. Bucarey, F. Ordóñez, and E. Bassaletti. Shape and balance in police districting. In *Applications of Location Analysis*, pages 329–347. Springer, 2015.

[9] H. Chen, W. Chung, J. J. Xu, G. Wang, Y. Qin, and M. Chau. Crime data mining: a general framework and some examples. *Computer*, 37(4):50–56, 2004.

[10] V. Conitzer and T. Sandholm. A technique for reducing normal-form games to compute a nash equilibrium. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 537–544. ACM, 2006.

[11] J. S. De Bruin, T. K. Cocx, W. Kosters, J. F. Laros, J. N. Kok, et al. Data mining approaches to criminal career analysis. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 171–177. IEEE, 2006.

[12] F. Fang, P. Stone, and M. Tambe. When security games go green: Designing defender strategies to prevent poaching and illegal fishing. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

[13] A. Gilpin and T. Sandholm. Better automated abstraction techniques for imperfect information games, with application to texas hold'em poker. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, page 192. ACM, 2007.

[14] A. Gilpin and T. Sandholm. Lossless abstraction of imperfect information games. *Journal of the ACM (JACM)*, 54(5):25, 2007.

[15] A. Gilpin, T. Sandholm, and T. B. Sørensen. A heads-up no-limit texas hold'em poker player: discretized betting models and automatically generated equilibrium-finding programs. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 2*, pages 911–918. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[16] J. P. Hespanha, M. Prandini, and S. Sastry. Probabilistic pursuit-evasion games: A one-step nash approach. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, volume 3, pages 2272–2277. IEEE, 2000.

[17] A. X. Jiang, Z. Yin, C. Zhang, M. Tambe, and S. Kraus. Game-theoretic randomization for security patrolling with dynamic execution uncertainty. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 207–214. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[18] J. Kalcsics, S. Nickel, and M. Schröder. Towards a unified territorial design approachâĂŤapplications, algorithms and gis integration. *Top*, 13(1):1–56, 2005.

[19] O. Kariv and S. L. Hakimi. An algorithmic approach to network location problems. ii: The p-medians. *SIAM Journal on Applied Mathematics*, 37(3):539–560, 1979.

[20] S. V. Nath. Crime pattern detection using data mining. In *Web Intelligence and Intelligent Agent Technology Workshops, 2006. WI-IAT 2006 Workshops. 2006 IEEE/WIC/ACM International Conference on*, pages 41–44. IEEE, 2006.

[21] T. H. Nguyen, F. M. Delle Fave, D. Kar, A. S. Lakshminarayanan, A. Yadav, M. Tambe, et al. Making the most of our regrets: Regret-based solutions to handle payoff uncertainty and elicitation in green security games. In *Decision and Game Theory for Security*, pages 170–191. Springer, 2015.

[22] T. Sandholm and S. Singh. Lossy stochastic game abstraction with bounds. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, pages 880–897. ACM, 2012.

[23] H. D. Sherali and F. L. Nordai. Np-hard, capacitated, balanced p-median problems on a chain graph with a continuum of link demands. *Mathematics of Operations Research*, 13(1):32–49, 1988.

[24] M. B. Short, M. R. D'ORSOGNA, V. B. Pasour, G. E. Tita, P. J. Brantingham, A. L. Bertozzi, and L. B. Chayes. A statistical model of criminal behavior. *Mathematical Models and Methods in Applied Sciences*, 18(supp01):1249–1267, 2008.

[25] M. Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge University Press, 2011.

[26] C. Zhang, A. X. Jiang, M. B. Short, P. J. Brantingham, and M. Tambe. Defending against opportunistic criminals: New game-theoretic frameworks and algorithms. In *Decision and Game Theory for Security*, pages 3–22. Springer, 2014.

[27] C. Zhang, A. Sinha, and M. Tambe. Keeping pace with criminals: Designing patrol allocation against adaptive opportunistic criminals. In *AAMAS 2015*.