Keeping Pace with Criminals: Designing Patrol Allocation Against Adaptive Opportunistic Criminals

Chao Zhang
University of Southern California
Los Angeles, CA 90089, USA
zhan661@usc.edu

Arunesh Sinha
University of Southern California
Los Angeles, CA 90089, USA
aruneshs@usc.edu

Milind Tambe
University of Southern California
Los Angeles, CA 90089, USA
tambe@usc.edu

ABSTRACT

Police patrols are used ubiquitously to deter crimes in urban areas. A distinctive feature of urban crimes is that criminals react opportunistically to patrol officers' assignments. Compared to strategic attackers (such as terrorists) with a well-laid out plan, opportunistic criminals are less strategic in planning attacks and more flexible in executing them. In this paper, our goal is to recommend *optimal* police patrolling strategy against such opportunistic criminals. We first build a game-theoretic model that captures the interaction between officers and opportunistic criminals. However, while different models of adversary behavior have been proposed, their exact form remains uncertain. Rather than simply hypothesizing a model as done in previous work, one key contribution of this paper is to learn the model from real-world criminal activity data. To that end, we represent the criminal behavior and the interaction with the patrol officers as parameters of a Dynamic Bayesian Network (DBN), enabling application of standard algorithms such as EM to learn the parameters. Our second contribution is a sequence of modifications to the DBN representation, that allows for a compact representation of the model resulting in better learning accuracy and increased speed of learning of the EM algorithm when used for the modified DBN. These modifications use marginalization approaches and exploit the structure of this problem. Finally, our third contribution is an iterative learning and planning mechanism that keeps updating the adversary model periodically. We demonstrate the efficiency of our learning algorithm by applying it to a real data set of criminal activity obtained from the police department of University of Southern California (USC) situated in Los Angeles, USA. We project a significant reduction in crime rate using our planning strategy as opposed to the actual strategy deployed by the police department. We also demonstrate the improvement in crime prevention in simulations when we use our iterative planning and learning mechanism compared to just learning once and planing. This work was done in collaboration with the police department of USC.

Categories and Subject Descriptors

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

General Terms

Algorithms, Human Factors, Security

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

Keywords

Game theory; Security games; Optimization

1. INTRODUCTION

Crime in urban areas plagues every city in all countries. A notable characteristic of urban crime, distinct from organized terrorist attacks, is that most urban crimes are opportunistic in nature, i.e., criminals do not plan their attacks in detail, rather they seek opportunities for committing crime and are agile in their execution of the crime [17, 20]. In order to deter such crimes, police officers conduct patrols with the aim of preventing crime. However, by observing on the spot the actual presence of patrol units, the criminals can adapt their strategy by seeking crime opportunity in less effectively patrolled location. The problem of where and how much to patrol is therefore important.

There are two approaches to solve this problem. The first approach is to determine patrol schedules manually by human planners, which is followed in various police departments including police in University of Southern California (USC). However, it has been demonstrated that manual planning of patrols is not only time-consuming but it is also highly ineffective in many related scenarios of protecting airport terminals [12] and ships in ports [16]. The second approach is to use automated planners to plan patrols against urban crime. This approach has either focused on modeling the criminal explicitly [20, 17] (rational, bounded rational, limited surveillance, etc.) in a game model or to learn the adversary behavior using machine learning [8]. However, the proposed mathematical models of criminal behavior have not been validated with real data. Also, prior machine learning approaches have only focused on the adversary actions ignoring their adaptation to the defenders' actions [8].

Hence, in this paper we tackle the problem of generating patrol strategies against opportunistic criminals. Our main novelty is in learning the criminal behavior from real data. We do so by modeling the interaction between the criminal and patrol officers as a Dynamic Bayesian Network (DBN). This DBN model is our *first contribution*. As far as we know, we are the first to use a DBN model that considers the temporal interaction between defender and adversary in the learning phase.

Given a DBN model, we can use the well-known Expectation Maximization (EM) algorithm to learn unknown parameters in the DBN from given learning data. However, using EM with the basic DBN model has two drawbacks: (1) the number of unknown parameters scales exponentially with the number of patrol areas and in our case is much larger than the available data itself; this results in over-fitting (2) EM cannot scale up due to the exponential growth of runtime in the number of patrol areas. We demonstrate these two drawbacks both theoretically and empirically.

Our *second contribution* is a sequence of modifications of the initial DBN model resulting in a compact representation of the model, that leads to better learning accuracy and increased speed of learning of the EM algorithm when used for the compact model. This sequence of modifications involve marginalizing states in the DBN using approximation technique from the Boyen-Koller algorithm [7] and exploiting structure of this problem. In the compact model, the parameters scale polynomially with the number of patrol areas, and EM applied to this compact model runs in polynomial time.

Our *third contribution* are two planning algorithms that enable computing the optimal officers' strategy. First, we present a dynamic programming based algorithm that computes the optimal plan in our planning and updating process. While the dynamic programming approach is optimal, it may be slow, hence we also present a fast but sub-optimal greedy algorithm to solve the planning problem. Further, the criminal behavior would change as he observes and reacts to the deployment of a new strategy. Hence, the optimal strategy with respect to the learnt behavior may not be effective for a long time, as the adversary behavior would have changed. Thus, we propose to frequently update our adversary model as we obtain new training data from a new deployment of defender strategy. By repeating the planning and updating process, we recommend officers' strategy that are more effective than learning just once.

Finally, as part of our collaboration with the police department of USC, we obtained criminal activity and patrol data for three years. This collaboration helped us validate our learning approach and also provided insights about the sequence of modifications that could be made for the basic DBN model. In fact, we project a significant reduction in crime rate using our approach as opposed to the current patrolling approach (see Figure 10). Given these results, we expect our algorithm to be tested and eventually deployed in USC. More broadly, by introducing a novel framework to reason about urban crimes along with efficient learning and planning algorithms, we open the door to a new set of research challenges.

2. RELATED WORK

We categorize the related work into five main areas. First, recent research has made inroads in applying machine learning and data mining in criminology domain to analyze crime patterns and support police in making decisions. A general framework for crime data mining is introduced in [8]. In [14], data mining is used to model crime detection problems and cluster crime patterns; in [9], data mining approaches are applied in criminal career analysis; in [15], the authors apply machine learning techniques to soft forensic evidence and build decision support systems for police. However, this area of research considers only crime data and does not model the interaction between patrol officers and criminals.

The second line of work we compare with is Pursuit-Evasion Games (PEG). PEG models a pursuer(s) attempting to capture an evader, often where their movement is based on a graph[11]. However, in common settings of Pursuit Evasion Games, evader's goal is to avoid capture and not to seek opportunities to commit crimes and a pursuer's goal is to capture the evader and not to deter the criminal; thus common PEG settings are different from the setting in this work.

The third area of work we compare with is Stackelberg Security Games (SSG) [18], which models the interaction between defender and attacker as a game and recommends patrol strategies for defenders against attackers. SSG has been successfully applied in security domains to generate randomized patrol strategies, e.g., to protect flights [18], for counter-terrorism and fare evasion checks on trains [13]. While the early work on SSG assumed a perfectly rational attacker, recent work has focused on attackers with bounded rationality and learning the parameters of the bounded rationality model using machine learning methods such as maximum-likelihood estimation. An example of this approach is the PAWS model [19]. PAWS addresses the problem of learning poacher behavior within a game-theoretic interaction between defenders and poachers.. Recent research has also made progress in designing patrol strategies against adversaries in graph settings [2]. In [3], patrol strategies against various types of adversaries are designed.

However, including various extensions, security games include an explicit model of the adversary such as bounded rationality models and limited observations models. In general, in security games, lack of sufficient data makes learning models of defender adversary interactions challenging. Distinct from these approaches, we do not model the adversary's decision-making explicitly, rather we learn the adversary interaction with defender using real world data. In our case these are how the adversary moves from one patrol area to another, and the his probability of committing a crime given some patrol officers presence.

A fourth thread of recent research combines machine learning with game theory. In [5], the defender's optimal strategy is generated in a SSG by learning the payoffs of potential attackers from their best responses to defender's deployments. An inherent problem with such an approach is that the defender strategy is geared towards learning the adversary payoff, and not exploiting the improved knowledge of the adversary payoff as the game progresses.

The last area of work we compare with is on modeling opportunistic criminals. In [17] burglars' movement is modeled as a random walk, and in [20], a more general model of opportunistic criminals was proposed with algorithms for optimal strategy against such criminals. Again, these papers include explicit models of the criminals and lack real world data to learn the interactions.

3. MOTIVATING EXAMPLE

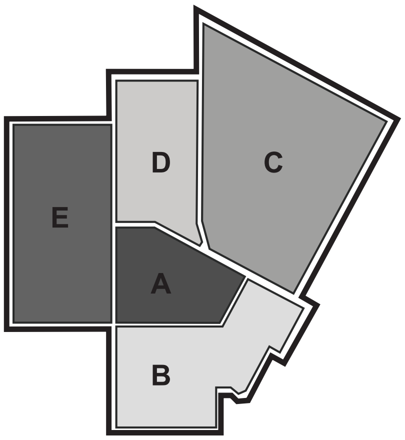


Figure 1: Campus map

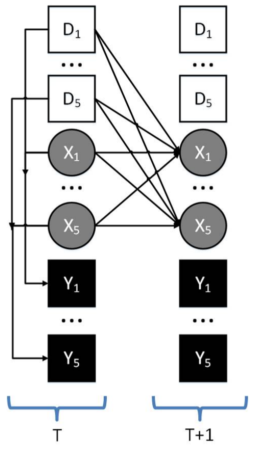
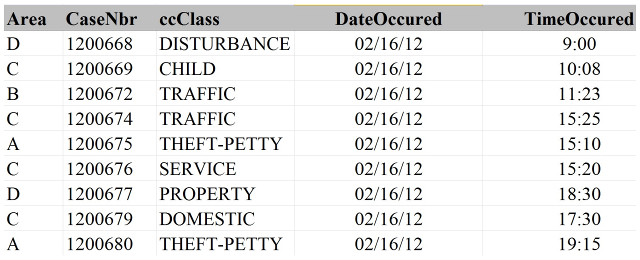


Figure 2: DBN for games

Domain Description: The motivating example for this study is the problem of controlling crime on a university campus. Our case study is about USC in USA. USC has a Department of Public Safety (DPS) that conducts regular patrols, similar to police patrols in urban settings. As part of our collaboration with USC DPS, we have access to the crime report as well as patrol schedule on campus for the last three years (2011-2013). USC is a large enough university that allows us to claim that our methods are applicable to other large campuses, including large mall areas.



Area	CaseNbr	ccClass	DateOccured	TimeOccured
D	1200668	DISTURBANCE	02/16/12	9:00
C	1200669	CHILD	02/16/12	10:08
B	1200672	TRAFFIC	02/16/12	11:23
C	1200674	TRAFFIC	02/16/12	15:25
A	1200675	THEFT-PETTY	02/16/12	15:10
C	1200676	SERVICE	02/16/12	15:20
D	1200677	PROPERTY	02/16/12	18:30
C	1200679	DOMESTIC	02/16/12	17:30
A	1200680	THEFT-PETTY	02/16/12	19:15

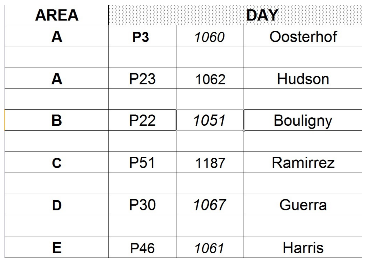
Figure 3: Sample Crime Report

In USC, the campus map is divided into five patrol areas, which is shown in Fig 1. DPS patrols in three shifts per day. In the crime data all crimes are local, i.e., no crime happens across two patrol areas or patrol shifts. At the beginning of each patrol shift, DPS assigns each available patrol officer to a patrol area and the officer patrols this area in this shift. At the same time, the criminal is seeking for crime opportunities by deciding which target they want to visit. Discussions with DPS reveal that criminals act opportunistically, i.e., crime is not planned in detail, but occurs when opportunity arise and there is insufficient presence of DPS officers.

Shift	A	B	C	D	E
1	1	1	2	2	2
2	1	1	1	2	1
3	2	1	1	3	1

Table 1: Crime data for 3 shifts.

There are two reports that DPS shared with us. The first is about criminal activity that includes details of each reported crime during the last three years, including the type of crime and the location and time information about the crime. We show a snapshot of this data in Figure 3. In this paper, we do not distinguish between the different types of crime and hence we consider only the number of crimes in each patrol area during each shift. Therefore, we summarize the three year crime report into $365 \times 3 \times 3 = 3285$ crime data points, one for each of the 8-hour patrol shift. Each crime data point contains five crime numbers, one for each patrol area.



AREA	DAY		
A	P3	1060	Oosterhof
A	P23	1062	Hudson
B	P22	1051	Bouligny
C	P51	1187	Ramirrez
D	P30	1067	Guerra
E	P46	1061	Harris

Figure 4: Patrol Schedule for 1 shift

The second data-set contains the DPS patrol allocation schedule. Every officer is allocated to patrolling within one patrol area. We show a snapshot of this data in Fig. 4. We assume that all patrol officers are homogeneous, i.e., each officer has the same effect on criminals' behavior. As a result, when generating a summary of officer patrol allocation data, we record only the number of officers allocated to each patrol area in each shift.

Table 2 shows a sample of the summarized officer patrol allocation data, where the row corresponds to a shift, the columns correspond to a patrol area and the numbers in each cell is the number of patrol officers. Table 1 shows a sample of the summarized crime data, where the row corresponds to a shift, the columns correspond to a patrol area and the numbers in each cell is the number of crimes. For example, from Table 2, we know that in shift 1, the number of officers in area A is 2 while the number of officers in area B, C, D and E is 1, while from Table 1 we know that in shift 1, there was 1 crime each in area A and B, and 2 crimes each in C, D and E. However, we do not know the number of criminals in any patrol area in any patrol shift. We call the patrol area as targets, and each patrol shift a time-step.

Problem Statement: Given data such as the real world data from USC, our goal is to build a general learning and planning framework that can be used to design optimal defender patrol allocations in any comparable urban crime setting.

Shift	A	B	C	D	E
1	2	1	1	1	1
2	1	1	2	2	2
3	2	1	1	3	1

Table 2: Patrol data for 3 shifts.

We model the learning problem as a DBN, and we describe the basic model and the EM algorithm in the next section. Then, we present a compact form of our model that leads to improved learning performance. After that, we present methods to find the optimal defender plan for the learnt model with frequent update of the criminal model.

4. LEARNING MODEL

We propose to learn the criminals' behavior, i.e, how the criminals pick targets and how likely are they to commit crime at that target. This behavior is in part affected by the defenders' patrol allocation. In this paper we assume that criminals are homogeneous, i.e., all criminals behave in the same manner. Further, as stated earlier, the patrol officers are also homogeneous. Thus, crime is affected only by the number of criminals and patrol officers, and not by which criminal or patrol officer is involved.

We propose a DBN model for learning the criminals' behavior. In every time-step of the DBN we capture the following actions: the defender assigns patrol officers to protect N patrol areas and criminals react to the defenders' allocation strategy by committing crimes opportunistically. Across time-steps the criminal can move from any target to any other, since a time-step is long enough to allow such a move. From a game-theoretic perspective, the criminals' payoff is influenced by the attractiveness of targets and the number of officers that are present. These payoffs drive the behavior of the criminals. However, rather than model the payoffs and potential bounded rationality of the criminals, we directly learn the criminal behavior as modeled in the DBN.

The DBN is shown in Fig 2: squares are observed states, where N white squares represent input states (number of defenders at each target) and N black squares represent output states (number of crime at each target) while N circles (number of criminals at each target) are hidden states. For ease of exposition, we use C to denote the largest value that any state can take. Next, we introduce the various parameters of this DBN.

4.1 DBN Parameters

First, we introduce parameters that measure size of the problem

- N: Total number of targets in the graph.
- T: Total time steps of the training data.

Next, we introduce random variables for the observed state (input defender distribution and output crime distribution in our case) and the hidden state. We use three random variables to represent the global state for defenders, criminals and crimes at all targets.

- d_t: Defender's allocation strategy at step t: number of defenders at each target in step t with C^N possible values.
- x_t: Criminals' distribution at step t with C^N possible values

- y_t: Crime distribution at step t with C^N possible values.

Next, we introduce the unknown parameters that we wish to learn.

- π: Initial criminal distribution: probability distribution of x_1.
- A (*movement matrix*): The matrix that decides how x_t evolves over time. Formally, $A(d_t, x_t, x_{t+1}) = P(x_{t+1}|d_t, x_t)$. Given the C^N values for each argument of A, representing A requires $C^N \times C^N \times C^N$ parameters.
- B (*crime matrix*): The matrix that decides how criminals commit crime. Formally, $B(d_t, x_t, y_t) = P(y_t|d_t, x_t)$. Given the C^N values for each argument of B, representing B requires $C^N \times C^N \times C^N$ parameters.

Next, we introduce variables that are used in the EM algorithm itself. These variables stand for specific probabilities as illustrated below. We use d_i^j (y_i^j) as shorthand for $d_i, \ldots, d_j$ ($y_i, \ldots, y_j$):
Forward prob.: $\alpha(k,t) = P(y_1^t, x_t = k|d_1^t)$,
Backward prob.: $\beta(k,t) = P(y_{t+1}^T|x_t = k, d_{t+1}^T)$,
Total prob.: γ: $\gamma(k,t) = P(x_t = k|y_1^T, d_1^T)$,
2-step prob.: $\xi(k,l,t) = P(x_t = k, x_{t+1} = l|y_1^T, d_1^T)$.

We can apply the EM algorithm to learn the unknown initial criminal distribution π, movement matrix A and output matrix B. However, EM applied to the basic DBN model above results in practical problems that we discuss in the next section.

4.2 Expectation Maximization

We start with a brief overview of EM. EM is a class of algorithms for finding maximum likelihood estimation for unknown parameters in DBN [10]. The EM algorithm has an initialization step, expectation (E) step and maximization (M) step. The initialization step chooses initial estimates for unknown parameters (π, A, B). The E step computes α, β, γ, ξ using these estimates. The M step updates the estimates of π, A, B using values of α, β, γ, ξ from E step. By iteratively performing E and M step, the EM algorithm converges to a local maxima of the likelihood function for parameters in the DBN. The particular mathematical equations used in E and M depends on the underlying model [4].

In EM algorithm, the size of movement matrix A is $C^N \times C^N \times C^N$ and the size of crime matrix B is also $C^N \times C^N \times C^N$. The number of unknown variables is $O(C^{3N})$. The exponentially many parameters make the model complex, and hence results in over-fitting given limited data. In addition, the time complexity as well as the space complexity of EM depends on the number of parameters, hence the problem scales exponentially with N. In practice, we can reduce C by categorizing the number of defenders, criminals and crimes. For example, we can partition the number of defenders, criminals and crimes into two categories each: the number of officers at each station is 1 (meaning ≤ 1) or 2 (meaning ≥ 2); the number of criminals/crimes is 0 (no criminal/ crime) or 1 (≥ 1 criminal/crime). However, the number of unknown parameters is still exponential in N. As a concrete example, in USC, $N = 5$ and the number of unknown parameters are more than 32768, even when we set $C = 2$. As we have daily data for three years, which is $365 \times 3 \times 3 = 3285$ data points, the number of parameters is much more than the number of data points. Therefore, we aim to reduce the number of parameters to avoid over-fitting and accelerate the computing process.

5. EM ON COMPACT MODEL (EMC^2)

In this section, we introduce our second contribution, which is to modify the basic DBN model to reduce the number of parameters. In the resultant compact model, the EM learning process runs faster and avoids over-fitting to the given data. The improvement may be attributed to the well-established learning principle of Occam's Razor [6], and our experimental results support our claims.

5.1 Compact model

We use three modifications to make our model compact. (1) We infer from the available crime data that crimes are local, i.e., crime at a particular target depends only on the criminals present at that target. Using this inference, we constructed a factored crime matrix B that eliminates parameters that capture non-local crimes. (2) Next, we rely on intuition from the Boyen-Koller [7] (BK) algorithm to decompose the joint distribution of criminals over all targets into a product of independent distributions for each target. (3) Finally, our consultations with the DPS in USC and prior literature on criminology [17] led us to conclude that opportunistic criminals by and large work independently. Using this independence of behavior of each criminal (which is made precise in Lemma 1), we reduce the size of the movement matrix. After these steps, the number of parameters is only $O(N \cdot C^3)$.

Before describing these modifications in details, we introduce some notations that aid in describing the different quantities at each target: $Y_t = [Y_{1,t}, Y_{2,t}, ..., Y_{N,t}]$ is a N by 1 random vector indicating the number of crimes $Y_{i,t}$ at each target i at step t. D_t is a N by 1 random vector indicating the number of defenders $D_{i,t}$ at each target i at step t. X_t is a N by 1 random vector indicating the number of criminals $X_{i,t}$ at each target i at step t.

Factored crime matrix: The number of crime at one target at one step is only dependent on the criminals and officers present at that target at that step. Therefore, we factor the crime matrix B to a matrix that has an additional dimension with N possible values, to represent how the criminals and officers at one target decide the crime at that target. Therefore, instead of the original crime matrix B of size $C^N \times C^N \times C^N$ matrix, we have a factored crime matrix of size $N \times C \times C \times C$ crime matrix. The first dimension of factored crime matrix represents the target, the second dimension represents the number of defenders at this target, the third dimension represents the number of criminals and the fourth dimension represents the number of crimes. We still refer to this factored crime matrix as B, where $B(i, D_{i,t}, X_{i,t}, Y_{i,t}) = P(Y_{i,t}|D_{i,t}, X_{i,t})$

Marginalized hidden state: The BK algorithm presents an approximation method by keeping the marginals of the distribution over hidden states, instead of the full joint distribution. Following the BK intuition, we marginalize the hidden state, i.e., instead of considering the full joint probability of criminals at all targets (with C^N possible values), we consider a factored joint probability that is a product of marginal probability of the number of criminals at each target.

In the unmodified DBN, the distribution over all the states at step t, $P(x_t)$ is a C^N by 1 vector. Additionally, the size of movement matrix A, which is the transition matrix from all the input and hidden state combinations at current step to the state at next step, is $C^N \times C^N \times C^N$. After marginalization, the marginals for each target i in the hidden state is $P(X_i = k, t)$, is a vector of size C. After we marginalize the hidden states, we only need to keep N marginals at each step, i.e., consider only N parameters. At each step, we can recover the distribution of full state by multiplying the marginals at this step. Then, we get the marginals at next step by evolving the recovered joint distribution of state at current step. Therefore, A can be expressed as a $C^N \times C^N \times N \times C$ matrix, where $A(d_t, x_t, i, X_{i,t+1}) = P(X_{i,t+1}|d_t, x_t)$.

Pairwise movement matrix A_m: Even with marginalized hidden state, we still need to recover the distribution of full state in order to propagate to next step. Therefore, the movement matrix size is still exponential with $C^N \times C^N \times N \times C$. In order to further reduce the number of unknown parameters and accelerate the computing process, we use properties of opportunistic criminals. Based on the crime reports and our discussion with DPS in USC, unlike organized terrorist attacks, the crimes on campus are committed by individual opportunistic criminals who only observe the number of defenders at the target they are currently at and do not communicate with each other. Therefore, at current step, the criminals at each target independently decide the next target to go to, based on their target-specific observation of number of defenders.

Based on the above observation, we can decompose the probability $P(X_{i,t+1} = 0|D_t, X_t)$ into a product of probabilities per target m. Denote by $X_{t+1}^{m \to i}$ the random variable that counts the number of criminals moving from target m to target i in the transition from time t to $t+1$. Lemma 1 proves that we can represent $P(X_{i,t+1} = 0|D_t, X_t)$ as a product of probabilities $P(X_{t+1}^{m \to i} = 0)$ for each m. $P(X_{t+1}^{m \to i} = 0)$ is a function of $D_{m,t}, X_{m,t}$

LEMMA 1. ***(Independence of behavior)*** *For a N target learning problem, given the number of defenders at each location $D_t = [D_{1,t}, ..., D_{N,t}]$ and the number of criminals $X_t = [X_{1,t}, ..., X_{N,t}]$, the probability $P(X_{i,t+1} = 0|D_t, X_t)$ of the number of criminal being 0 at location i at step $t+1$ is given by $\prod_{j=1}^N P(X_{t+1}^{j \to i} = 0)$.*

PROOF 1. *Note that we must have $X_{t+1}^{m \to i} \geq 0$. We have the total number of criminals at target i at time step $t+1$ as $X_{i,t+1} = \sum_m X_{t+1}^{m \to i}$, i.e, the number of criminals at target i at step $t+1$ is the sum of criminals that move from each target to target i. Clearly $X_{i,t+1} = 0$ iff $X_{i,t+1}^{D_{m,t},X_{m,t}} = 0$. Therefore, we have $P(X_{i,t+1} = 0|D_t, X_t) = P(X_{t+1}^{1 \to i} = 0, \dots, X_{t+1}^{N \to i} = 0)$. Since the criminals' decisions at each target are independent, we have $P(X_{t+1}^{1 \to i} = 0, ..., X_{t+1}^{N \to i} = 0) = \prod_{m=1}^N P(X_{t+1}^{m \to i} = 0)$.*

When $C = 2$ and $X_{i,t} \in \{1, 2\}$, we can construct the whole movement matrix A using $P(X_{t+1}^{m \to i} = 0)$ (pairwise transition probabilities) by utilizing the fact that $P(X_{i,t+1} = 1|D_t, X_t) = 1 - P(X_{i,t+1} = 0|D_t, X_t)$. Therefore, instead of keeping A, we keep a transition matrix A_m where $A_m(i, D_{i,t}, X_{i,t}, j, X_{j,t+1}) = P(X_{t+1}^{i \to j})$. The number of parameters in A_m is $N \times 2 \times 2 \times N = 4N^2$. We do not consider the range of $X_{j,t+1}$ because we only need one parameter to store the two cases of $X_{j,t+1} = 1$ and $X_{j,t+1} = 0$ since $A_m(i, D_{i,t}, X_{i,t}, j, X_{j,t+1} = 1) = 1 - A_m(i, D_{i,t}, X_{i,t}, j, X_{j,t+1} = 0)$. When $C > 2$, the number of variables in A_m are $C^2(C-1)N^2$; we can readily extend the above construction of A from A_m, which we show in the appendix (https://dl.dropboxusercontent.com/u/98294554/Appendix.pdf).

5.2 EMC² procedure

EM on CompaCt model (EMC^2) procedure applies the EM algorithm to the compact DBN model. To learn the initial distribution $\pi_{k,i} = P(X_{i,1} = k)$, matrix A_m and matrix B, we first generate initial estimates of these parameters that satisfy the condition $\sum_k \hat{\pi}(k,i) = 1$, $\sum_{X_{j,t+1}} \hat{A}_m(i, D_{i,t}, X_{i,t}, j, X_{j,t+1}) = 1$ and $\sum_{Y_{i,t}} \hat{B}(i, D_{i,t}, X_{i,t}, Y_{i,t}) = 1$.

Next, we define the intermediate variables used in the EM algorithm. These differ from the earlier application of EM because of our changed model. We use the shorthand Y_i^j to denote $Y_i, ..., Y_j$ and D_i^j to denote $D_i, ..., D_j$:
Forward prob.: $\alpha(i,k,t) = P(Y_1^t, X_{i,t} = k|D_1^t)$,
Backward prob.: $\beta(i,k,t) = P(Y_{t+1}^T|X_{i,t} = k, D_t^T)$,
Total prob.: $\gamma(i,k,t) = P(Y, X_{i,t} = k|D_1^T)$,
2-step prob.: $\xi(i,k,j,l,t) = P(X_{i,t} = k, X_{j,t+1} = l|Y_1^T, D_1^T)$.

Next, the E and M steps are used with random restarts to learn the values of π, A_m and B. While the equations used in the E and M steps can be derived following standard EM techniques, we illustrate a novel application of the distributive law for multiplication in the E step that enables us to go from exponential time complexity to polynomial (in N) time complexity. Without going into details of the algebra in the E step, we just focus on the part of the E step that requires computing $P(Y_1^{t-1}, X_{i,t} = 0|D_1^t)$.

The following can be written from total law of probability

$$P(Y_1^{t-1}, X_{i,t} = 0|D_1^t) = \sum_{X_{t-1}} P(Y_1^{t-1}, X_{i,t} = 0, X_{t-1}|D_1^t)$$
$$= \sum_{X_{t-1}} P(Y_1^{t-1}|D_1^t, X_{i,t} = 0, X_{t-1}) P(X_{i,t} = 0|D_1^t, X_{t-1})$$
$$P(X_{t-1}|D_1^t)$$

The above can be simplified using the Markovian assumptions of the DBN to the following

$$\sum_{X_{t-1}} P(Y_1^{t-1}|D_1^t, X_{t-1}) P(X_{i,t} = 0|D_{t-1}, X_{t-1}) P(X_{t-1}|D_1^t)$$

The first and third term can be combined (Bayes theorem) to obtain

$$\textstyle\sum_{X_{t-1}} P(Y_1^{t-1}, X_{t-1}|D_1^t) P(X_{i,t} = 0|D_{t-1}, X_{t-1})$$

Using the Boyen-Koller assumption in our compact model we get

$$P(Y_1^{t-1}, X_{t-1}|D_1^t) = \textstyle\prod_j P(Y_1^{t-1}, X_{j,t-1}|D_1^t)$$

Also, using Lemma 1 we get

$$P(X_{i,t} = 0|D_{t-1}, X_{t-1}) = \textstyle\prod_j P(X_t^{j \to i} = 0)$$

Thus, using these we can claim that $P(Y_1^{t-1}, X_{i,t} = 0|\mathcal{D}_t)$ is

$$\textstyle\sum_{X_{t-1}} \prod_j P(Y_1^{t-1}, X_{j,t-1}|D_1^t) P(X_t^{j \to i} = 0)$$

Since the range of X_{t-1} is C^N, naively computing the above involves summing C^N terms, thus, implying a time complexity of $O(C^N)$. The main observation that enables polynomial time complexity is that we can apply principles of the generalized distributive law [1] to reduce the computation above. As an example, the three summations and four multiplication in $ab+ac+bc+bd$ can be reduced to two summations and one multiplication by expressing it as $(a+b)(c+d)$. Using distributive law we reduce the computation for $P(Y_1^{t-1}, X_{i,t} = 0|D_1^t)$ by switching sum and product

$$\textstyle\prod_j \sum_{X_{j,t-1}} P(Y_1^{t-1}, X_{j,t-1}|D_1^t) P(X_t^{j \to i} = 0)$$

The complexity of computing the above is $O(N^C)$. Applying this idea, we can calculate α, β, γ and ξ from the estimated value of $\hat{\pi}$, $\hat{A_m}$ and $\hat{B}$ in the expectation step in time polynomial in N.

For the maximization step, we update the estimate of π, A_m and B using the probabilities we derive in the expectation step. The procedure is the same as Equation (7) to (9), hence we provide the details in the appendix.

Computational complexity analysis The complexity of EM on the basic model is $O(C^{2N}T)$, and for EMC^2 it is $O(N^{C+1}T + (C \cdot N)^2 T)$. The detailed derivation is not hard and delegated to the appendix. Therefore, EMC^2 procedure runs much faster than EM in the basic model when C is small.

6. DYNAMIC PLANNING

The next step after learning the criminals' behavior is to design effective officer allocation strategies against such criminals. In this section, we first introduce a simple online planning mechanism,

Algorithm 1 Online planning ($Train\_data, T_u, T$)

```
1: $A, B, \pi \leftarrow Learn(Train\_data)$
2: $t = 0$
3: while $t < T$ do
4:     $[D_1, ..., D_{T_u}] \leftarrow Plan(A, B, \pi)$
5:     $[Y_1, ..., Y_{T_u}] \leftarrow Execute\{D_1, ..., D_{T_u}\}$
6:     $Train\_data \leftarrow Train\_data \cup \{D_1, Y_1, ..., D_{T_u}, Y_{T_u}\}$
7:     $A, B, \pi \leftarrow Update(Train\_data, A, B, \pi)$
8:     $t = t + T_u$
9: end while
```

in which we iteratively update criminals' behavior model and plan allocation strategies. Next, we present a slower optimal planning algorithm and faster but sub-optimal greedy algorithm.

Online Planning Mechanism. We first state our template for iterative learning and planning before describing the planning algorithms. The criminal behavior may change when the criminal observes and figures out that the defender strategy has changed. Thus, the optimal strategy planned using the learned parameters is no longer optimal after some time of deployment of this strategy, as the parameters itself change in response to the deployed strategy.

To address the problem above, we propose an online planning mechanism. In this mechanism, we update criminal's model based on real-time crime/patrol data and dynamically plan our allocation strategy. The first step is to use the initial training set to learn an initial model. Next, we use a planning algorithm to generate a strategy for the next T_u steps. After executing this strategy, we can collect more crime data and use them to update the model with the original training data. By iteratively doing this, we generate strategies for the whole horizon of T steps. Algorithm 1 presents the details of this mechanism.

Compared to simply applying planning algorithm for T steps, our online planning mechanism updates criminals' behavior model periodically based on his response to the currently deployed strategy. In this online planning mechanism, three parts are needed: learning algorithm, updating algorithm and planning algorithm. For learning and updating algorithm, we apply the EMC^2 learning algorithm from Section 5. In addition, we also need a planning algorithm, which we discuss next.

6.1 Planning Algorithms

The planning problem. In the planning problem, the criminals' behavior is known, or more specifically, we already know the criminals' initial distribution π, movement matrix A and crime matrix B in the DBN model. Given a pure defender patrol allocation strategy for T_u steps, we can plug those values for the input state in the DBN and get the expected number of crimes in T_u steps. The goal of planning is to find the defenders' pure strategy that optimizes the defenders' utility, which in our case is to minimize the total expected number of crimes. (In our way our framing, any randomized strategy, which is the combination of pure strategies, results in more number of crimes than the optimal pure strategy). Thus, planning against opportunistic criminals is a search problem in defender's pure strategy space. First, we present the practical impossibility of a brute force search.

Brute Force search: A naive way to solve this problem is to try all possible allocation strategies and pick the one that leads to least crimes in T_u steps. However, since at each step, the number of possible allocation strategies is C^N and there are T_u steps in total, the strategy space is C^{NT_u}. For example, for our specific problem of patrolling in USC with five targets, two categories and the goal of planning for $T_u = 300$ steps, we need to search $2^{1500} \approx 10^{451}$ different strategies, which is impractical to solve.

Algorithm 2 DOGS (A, B, π)

```
1: for each officer allocation $D_1^i$ do
2:     $\mathsf{Pa}[i, 1] \leftarrow 0$;  $P_{i,1} \leftarrow f_Y(A, \pi, D_1^i)$;  $X_{i,1} \leftarrow \pi$
3: end for
4: for $t \leftarrow 2, 3, ..., T_u$ do
5:     for each officer allocation $D_t^j$ do
6:         $F(i) = f_Y(f_X(A, X_{i,t-1}, D_{t-1}^i), D_t^j, B) + P_{i,t-1}$
7:         $\mathsf{Pa}[D_t^j, t] \leftarrow \text{argmin}_i[F(i)]$; $P_{j,t} \leftarrow \min_i[F(i)]$
8:         $X_{j,t} \leftarrow f_X(A, X_{\mathsf{Pa}[D_t^j,t],t-1}, D_{t-1}^{\mathsf{Pa}[D_t^j,t]})$
9:     end for
10: end for
11: $index[T] \leftarrow \text{argmin}_i P_{i,T}$;  $\widehat{D}[T] \leftarrow D_T^{index[T]}$
12: for $t \leftarrow T-1, ..., 1$ do
13:     $index[t] \leftarrow \mathsf{Pa}[D_t^{index[t+1]}, t+1]$
14:     $\widehat{D}[t] \leftarrow D_t^{index[t]}$
15: end for
16: return $\widehat{D}$
```

Dynamic Opportunistic Game Search (DOGS): First, we list some notation that will be used in the next two planning algorithms.

- D_t^j indicates the j^{th} strategy for the defender from the C^N different defender strategies at time step t.
- $P_{j,t}$ is the total number of crimes corresponding to the optimal defender strategy for the first t time-steps that has j as its final defender strategy.
- $X_{j,t}$ is the criminals' location distribution corresponding to the optimal defender strategy for the first t time-steps that has j as its final defender strategy.
- $f_Y(X_t, D, B)$ is the expected number of crimes at all targets at t given the criminal location distribution X_t and defender's allocation strategy D at step t and output matrix B.
- $f_X(A, X_t, D_t)$ is the criminal location distribution at step $t+1$ given the criminal location distribution X_t and defender's allocation strategy D_t at t and transition matrix A.

DOGS is a dynamic programming algorithm, hence in order to find the optimal strategy for t steps, we first find the optimal strategy for the sub-problem with $t-1$ steps and use it to build the optimal strategy for t steps. Given the values of π, A and B from our learning step, the optimal defender allocation strategy $D_1, ..., D_{T_u}$ is given by the recurrence relations:

$$P_{j,1} = f_Y(\pi, D_1^j, B)$$
$$P_{j,t} = \min_i[f_Y(f_X(A, X_{i,t-1}, D_{t-1}^i), D_t^j, B) + P_{i,t-1}]$$

Retrieving the optimal allocation strategy requires remembering the allocation D_{t-1}^i that minimizes the second equation, which is done by storing that information in the function Pa, as follows:

$$\mathsf{Pa}[j, t] = \underset{i}{\text{argmin}}[f_Y(f_X(A, X_{i,t-1}, D_{t-1}^i), D_t^j, B) + P_{i,t-1}]$$

As P_{j,T_u} is the total number of crime for the optimal defender strategies for T_u time-steps that has j as the final strategy, the optimum strategy for time-step T_u is given by $D_{T_u} = \text{argmin}_j P_{j,T_u}$. Then, recursively, given optimal D_t we find the optimal strategy in the previous time-step using function Pa: $D_{t-1} = \mathsf{Pa}[D_t, t]$. The complexity of DOGS algorithm (Algorithm 2) is $O(C^{2N}T_u)$.

Greedy search. The dynamic programming based algorithm can generate the optimal strategy, but takes time $O(C^{2N}T_u)$. We present a greedy algorithm that runs in $O(C^N T_u)$ time, but the solution may be sub-optimal. In greedy search, we split the strategy space into T_u slices. Each slice represents the strategy at each

Algorithm 3 GREEDY $(A, B, \pi = X_1)$

```
1: for $t \leftarrow 1, \ldots, T_u$ do
2:     $D_t \leftarrow \text{argmin}_D f_Y(X_t, D, B)$; $X_{t+1} \leftarrow f_X(A, X_t, D_t)$
3: end for
4: return $D = [D_1, ..., D_{T_u}]$
```

step. Then, instead of searching the optimal strategy for T_u steps, we only look one step ahead to search the strategy that optimize defender's utility at current step (Algorithm 3). It finds the optimal patrol allocation D_t at current step by minimizing the expected number of crime at all targets at step t. For the next step, we compute the criminal's distribution X_{t+1} and greedily search again. We keep iterating this process until we reach T_u step. The complexity of Greedy search is $O(C^N T_u)$.

7. EXPERIMENTAL RESULTS

Experimental setup. All our experiments were performed on a machine with 2.4GHz and 16GB RAM. MATLAB was our choice of programming language. There are two threads of experiments, one on learning and other on learning and planning. To avoid leaking confidential information of USC Department of Public Safety, all the crime numbers shown in the results are normalized.

Learning (Setting): Our first experiment is on evaluating performance of EMC^2 algorithm in learning criminals' behavior. We use the case study of USC in our experiments. We obtained three years of crime report and corresponding patrol schedule followed in USC. Since EMC^2 algorithm and EM algorithm only reach locally optimal solution, we run the algorithms for 30 different randomly chosen start points and choose the best solution from among these runs. These start points, i.e., values of A, B and π, are generated by sampling values from a uniform random distribution over $[0, 1]$ for all the elements and then normalizing the probabilities so that they satisfy the initial conditions. C is set to 2 by default while the effect of varying C is compared in Figure 9.

Results: The results shown in Figure 5 compares the estimated numbers of crimes using different learning algorithms with real number of crimes in 30 days. Three different algorithms are compared: (1) the Markov chain (MC) algorithm, in which the problem is modeled as a Markov chain where the states represent the number of defenders and crimes at all targets, (2) the exact EM algorithm and (3) the EMC^2 algorithm. We divide the three year data into four equal parts of nine months each. For each part we train on the first eight months data and test on the ninth month data. The x-axis in this figure indicates the index of the part of data that we evaluate on. y-axis is the total number of crimes in 30 days. The closer this number is to the real number of crime, the better the prediction is. As can be seen, the prediction of EMC^2 is much closer compared to those of EM and MC algorithm in all the training groups. This indicates that the crime distribution is related to criminals' location and including number of criminals at each target as a hidden state helps improving performance. In addition, EMC^2 algorithm achieves better performance than EM by reducing number of unknown variables to avoid over-fitting.

For Figure 6, we measure learning performance for each individual target using a metric that we call accuracy. To define this metric, let n_{it} be the actual number of crimes at target i for time step t, let n'_{it} be the predicted number of crimes at target i at time step t. Then, accuracy at step t is the probability of the event $\sum_{i=1}^{N} |n_{it} - n'_{it}| \leq 1$. In other words, it is the probability that we make less than one mistake in predicting crimes for all N targets. The reported accuracy is the average accuracy over all t. In Figure 6, the y-axis represents the accuracy. The higher accuracy is, the more accurate our prediction is. We compare four different algorithm: MC, EM, EMC^2 algorithm and the uniform random algorithm, which sets equal probability for all possible numbers of crimes at each target. As expected, EMC^2 outperforms all other algorithms in all training groups. In addition, even though the accuracy of the algorithms varies in different training groups, which we attribute to the noisy nature of the data in the field, the largest difference is within 15%. This indicates accuracy of the algorithms are data-independent.

We present additional results under this setting in Figure 7 and 8. We compare the four approaches for varying size of training data, thus, the x-axis in both figures shows the number of training data (in days of data) used in learning. Our test data is all of the data points from a 30 day period, and the training data are the data points just before (in order of time) the test data points. For Figure 7, EMC^2 algorithm again outperforms all other algorithms for any number of training data in accuracy. In addition, the more data we have for training, the better accuracy we achieve. In Figure 8, the y-axis shows runtime in seconds on a log scale. The more data we have, the longer it takes for each training method. Random algorithm is pre-generated and takes almost no time, hence that data is not shown in the figure; the runtime for MC is negligible because the number of state is small ($O(4^N)$) and we traverse all the data points only once; the runtime for EMC^2 algorithm is significantly better than that for EM algorithm, as is expected by our complexity analysis in Section 5.

In Figure 9, we compare the four approaches by varying C. The x-axis shows the value of C. We use 1100 data points for training while 30 data points, which is just after the training data points, are used for testing. The accuracy decreases as C increases. This is because when C increases, there are more possible values of number of crimes. Thus, the possibility of predicting an accurate number decreases. However, when C increases from 3 to 4, the decrease in accuracy is small in EMC^2 due to the fact that data with value 4 rarely appears in both the crime and patrol data-set. This indicates a small C is a good approximation. In addition, EMC^2 algorithm again outperforms all other algorithms for any C.

Learning and Planning (Real world data): Figure 10 compares DOGS with the actual deployed allocation strategy generated by DPS experts in USC. Similar to the settings in Figure 5, we divide the three year data into four equal parts of nine months. For each part we train on the first eight months data using EMC^2 algorithm and test different allocation strategy on the first 10 days of the ninth month data. When testing the strategy, we assume the criminals' behavior remain unchanged during these 10 days. Three different scenarios are compared: (1) the real number of crimes, shown as Real in Fig. 10; (2) the expected number of crimes with DPS strategy and learned criminal behavior, shown as Real-E and (3) the expected numbers of crime with DOGS allocation and learned criminal behavior, shown as DOGS. As shown in Fig 10, the expected number of crime with DPS strategy is close to the real number of crimes, which indicates EMC^2 captures the main features of the criminal behavior and provides close estimate of the number of crimes. In addition, DOGS algorithm outperforms the strategy generated by domain experts significantly. This demonstrates the effectiveness of DOGS algorithm as compared to current patrol strategy. By using allocation strategy generated by DOGS, the total crime number reduces by $\sim$50% as compared to the currently deployed strategy.

Learning and planning (Simulated data): Next, we evaluate the performance of our online planning mechanism. We use simulations for this evaluation. In the simulation, the criminal model is simulated using the model from an earlier work on opportunistic criminals [20], in which the authors explicitly model an opportunis-

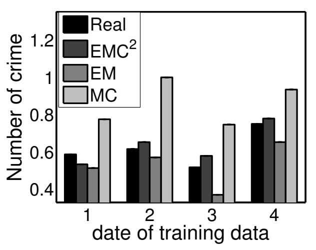


Figure 5: Total number of crime

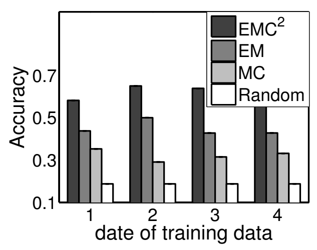


Figure 6: Individual Accuracy

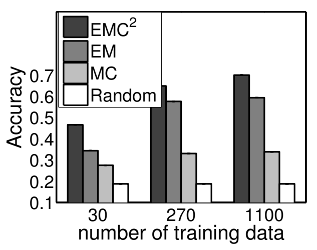


Figure 7: Varying data

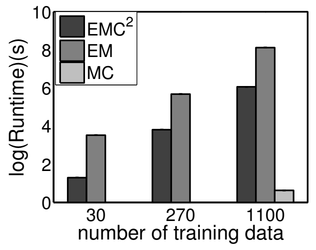


Figure 8: Varying data(Runtime)

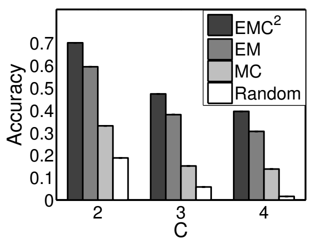


Figure 9: Vary C

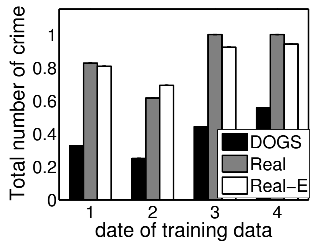


Figure 10: Compare with deployed

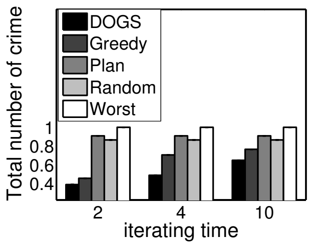


Figure 11: Vary T_u

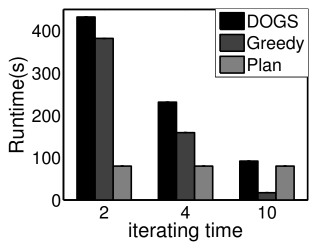


Figure 12: Vary T_u(Runtime)

tic criminal's behavior. However, the defender does not know the type of criminals in our experiments. Instead, the defender starts by executing a random patrol schedule for 300 steps and collects the corresponding crime report using which they learn an initial criminal behavior model. The criminal responds to the defenders' patrol schedule as predicted by the behavior model in [20]. Since the criminal behavior in [20] is probabilistic, we run the experiment 30 times and each data point we report in this part is an average over these 30 instances. We fix the number of patrol officers to $2N-2$, where N is the number of targets. This number is consistent with our real data-set numbers (8 officers for 5 targets), where there were enough officers to allocate one officer to each target, but not enough to allocate two officers to each target. We use EMC^2 algorithm as the learning algorithm.

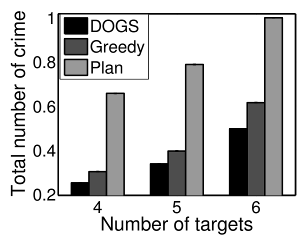


Figure 13: Varying N

Results: Figure 11 to 13 presents the results from our experiments about the online learning and planning mechanism. Four planning mechanisms that we consider are as follows: first, a random planning mechanism that randomly generates allocation strategy with limited resources; second, a pure planning mechanism, where we learn the criminal behavior model once and apply this model to plan for the entire horizon T using DOGS algorithm; third, a online planning mechanism with greedy planning algorithm that updates every T_u time-steps; and the last mechanism is online planning mechanism with DOGS algorithm that also updates every T_u time-steps. In Figure 11, the total planning horizon T is set to 600. In addition to the four planning mechanisms, we also consider the worst case where the defender always protect the least valuable targets. The x-axis shows the update interval T_u, which is the time interval after which we update criminals' behavior model. The y-axis is the expected number of crimes that happens under the deployed allocation strategy within 600 steps. Expected number of crimes under pure planning mechanism stay the same with different T_u because it does not update the criminals' model at all. For online mechanisms, the expected number of crimes increases as the update interval T_u increases. This is because with infrequent updates of the criminals' behavior model, we cannot keep up with the real criminals' behavior. In addition, with any size of the update interval, DOGS algorithm outperforms the greedy algorithm. In Figure 12, we present the runtime of three mechanisms for the same experiment. We do not show the runtime for the random planning mechanism as it is small and same for any planning horizon T. The runtime decreases as the update interval T_u increases. There is a runtime-quality trade-off in choosing T_u. Figure 13 shows the performance of the four planning mechanisms, but with different number of targets in the model. The x-axis is the number of targets in the graph and the y-axis is the expected number of crimes under the deployed strategy. We set $T = 600$, $T_u = 2$. The results here are similar to the results of Fig. 11.

These results lead us to conclude that online mechanisms outperform the baseline planning mechanisms significantly in any settings. For online mechanisms, DOGS achieves better performance while greedy planning algorithm requires less runtime. Thus, based on the specific problem being solved, the appropriate algorithm must be chosen judiciously.

8. CONCLUSION

This paper introduces a novel framework to design patrol allocation against adaptive opportunistic criminals. First, we model the interaction between officers and adaptive opportunistic criminals as a DBN. Next, we propose a sequence of modifications to the basic DBN resulting in a compact model that enables better learning accuracy and running time. Finally, we present an iterative learning and planning mechanism with two planning algorithm to keep pace with adaptive opportunistic criminals. Experimental validation with real data supports our choice of model and assumptions. Further, our modeling assumptions were informed by inputs from our collaborators in the DPS at USC. These promising results have opened up the possibility of deploying our method in USC. This paper has further opened up the integration of opportunistic crime security games [20] with machine learning.

9. ACKNOWLEDGEMENT

This research is supported by MURI grant W911NF-11-1-0332. We thank Kevin Knight for many helpful comments on this work.

REFERENCES

[1] S. M. Aji and R. J. McEliece. The generalized distributive law. *Information Theory, IEEE Transactions on*, 46(2):325–343, 2000.

[2] 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.

[3] 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.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[5] A. Blum, N. Haghtalab, and A. D. Procaccia. Learning optimal commitment to overcome insecurity. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NIPS). Forthcoming*, 2014.

[6] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Inf. Process. Lett.*, 24(6):377–380, Apr. 1987.

[7] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 33–42. Morgan Kaufmann Publishers Inc., 1998.

[8] 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.

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

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[11] 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.

[12] M. Jain, J. Tsai, J. Pita, C. Kiekintveld, S. Rathi, M. Tambe, and F. Ordóñez. Software assistants for randomized patrol planning for the lax airport police and the federal air marshal service. *Interfaces*, 40(4):267–290, 2010.

[13] 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.

[14] 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.

[15] G. Oatley, B. Ewart, and J. Zeleznikow. Decision support systems for police: Lessons from the application of data mining techniques to soft forensic evidence. *Artificial Intelligence and Law*, 14(1-2):35–100, 2006.

[16] E. Shieh, B. An, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, and G. Meyer. Protect: A deployed game theoretic system to protect the ports of the united states. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 13–20. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[17] 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.

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

[19] R. Yang, B. Ford, M. Tambe, and A. Lemieux. Adaptive resource allocation for wildlife protection against illegal poachers. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 453–460. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[20] 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.