Journal of Artificial Intelligence Research 50 (2014) 321-367 Submitted 1/14; published 6/14

Game-Theoretic Security Patrolling with Dynamic Execution Uncertainty and a Case Study on a Real Transit System

Francesco M. Delle Fave DELLEFAV@USC.EDU
Albert Xin Jiang JIANGX@USC.EDU
Zhengyu Yin ZHENGYUY@USC.EDU
Chao Zhang ZHAN661@USC.EDU
Milind Tambe TAMBE@USC.EDU
University of Southern California,
Los Angeles, CA 90089 USA

Sarit Kraus SARIT@CS.BIU.AC.IL
Bar Ilan University,
Ramat Gan 52900, Israel

John P. Sullivan JPSULLIV@LASD.ORG
Los Angeles County Sheriff's Department
Los Angeles, CA 90059

Abstract

Attacker-Defender Stackelberg security games (SSGs) have emerged as an important research area in multi-agent systems. However, existing SSGs models yield fixed, static, schedules which fail in dynamic domains where defenders face execution uncertainty, i.e., in domains where defenders may face unanticipated disruptions of their schedules. A concrete example is an application involving checking fares on trains, where a defender's schedule is frequently interrupted by fare evaders, making static schedules useless.

To address this shortcoming, this paper provides four main contributions. First, we present a novel general Bayesian Stackelberg game model for security resource allocation in dynamic uncertain domains. In this new model, execution uncertainty is handled by using a Markov decision process (MDP) for generating defender policies. Second, we study the problem of computing a Stackelberg equilibrium for this game and exploit problem structure to reduce it to a polynomial-sized optimization problem. Shifting to evaluation, our third contribution shows, in simulation, that our MDP-based policies overcome the failures of previous SSG algorithms. In so doing, we can now build a complete system, that enables handling of schedule interruptions and, consequently, to conduct some of the first controlled experiments on SSGs in the field. Hence, as our final contribution, we present results from a real-world experiment on Metro trains in Los Angeles validating our MDP-based model, and most importantly, concretely measuring the benefits of SSGs for security resource allocation.

1. Introduction

In recent years, research in algorithmic game theory has started to show a significant interest in security resource optimization problems. This research has led to decision aids for real-world security agencies which need to deploy patrols and checkpoints to protect targets from terrorists and criminals (Tambe, 2011). Stackelberg security games (SSGs) have been advocated as a powerful tool to model these problems (Gatti, 2008; Conitzer, 2012; Basilico, Gatti, & Amigoni, 2009a; Vorobeychik & Singh, 2012; Vanek, Jakob, Lisy, Bosansky, & Pechoucek, 2011; Pita, Jain, Western, Portway, Tambe, Ordonez, Kraus, & Paruchuri, 2008; Tambe, 2011). An SSG is a two-player game between a defender (the security agency) and an adversary (a terrorist or a criminal). The defender commits to a mixed strategy— a randomized resource allocation specified by a probability distribution over deterministic schedules— which takes into the account the adversary's best response to his observation of the mixed strategy[1]. Several decision-support systems based on SSGs have been successfully deployed in real world domains for assisting the security of ports, airports, ferries and transit systems. Examples include ARMOR and GUARDS for airport security (Pita et al., 2008; Pita, Tambe, Kiekintveld, Cullen, & Steigerwald, 2011), IRIS for allocating security personnel to international flights of US Carriers (Tsai, Rathi, Kiekintveld, Ordóñez, & Tambe, 2009), PROTECT for randomized patrols for security of ports and passenger ferries in ports such as New York, Boston and Los Angeles (Shieh, An, Yang, Tambe, Baldwin, DiRenzo, Maule, & Meyer, 2012; Fang, Jiang, & Tambe, 2013) and TRUSTS for patrolling Metro trains in Los Angeles (Yin, Jiang, Johnson, Tambe, Kiekintveld, Leyton-Brown, Sandholm, & Sullivan, 2012).

Some of the domains discussed above involve patrolling a transportation system such as a train-line, a ferry or a flight system. In such settings, schedules are typically time-critical because they depend on the time table of the vehicles (trains, ferries or flights). However, interruptions are frequent while patrolling key transportation systems because the officer might have to respond to an emergency, provide assistance to a passenger or need to arrest someone. For example, when patrolling trains, whenever an officer is delayed midway, it might become impossible for the officer to complete his patrol schedule. Hence, *fixed* schedules that cannot be updated after an interruption will be hard to follow after an officer is delayed. Unfortunately, previous work has often provided static or fixed patrolling schedules that face problems in the presence of unanticipated disruptions.

In general, such *execution uncertainty* is endemic in transportation domains and it will affect the defender units' ability to carry out their planned schedules in later time steps. One motivating example, which will be used throughout this work, is the TRUSTS system for scheduling fare inspections in the Los Angeles metro rail system (LA Metro). TRUSTS (Yin et al., 2012), currently being evaluated by the Los Angeles sheriff's department (LASD), provides a game-theoretic solution to scheduling randomized patrols for fare inspections on trains and at stations. As we will see later in this paper, in real world trials carried out by the LASD, a significant fraction of the executions of the pre-generated schedules got interrupted for a variety of reasons such as writing citations, felony arrests, and handling emergencies. Such interruptions caused the officers to miss the train that they were supposed to take as part of their patrol schedule. On such occasions, the solution of TRUSTS did not provide instructions on what to do after the interruption making the schedules useless to the officers.

Previous work has addressed some aspects of execution uncertainty. In particular, Yin, Jain, Tambe, and Ordonez (2011), Yin and Tambe (2012) present two different approaches, one based on robust optimization and another on a Bayesian method, whereby the defender optimizes her mixed strategy taking into account that some (small) fraction of it will be

1. By convention in security games literature, the defender is referred to as "she" and the adversary as "he".

incorrectly executed. Unfortunately, as discussed above, in the domains of interest in this work, including TRUSTS, a significant fraction of the schedules are interrupted. Most importantly, in such cases, this previous work does not suggest any alternatives whenever the defenders on patrol are interrupted—thus it fails to optimally use the patrol time. This clearly indicates that a key challenge still needs to be addressed in SSGs: a new framework is needed, that can generate patrol schedules that are robust against execution uncertainty and that can provide contingency plans whenever disruptions occur.

To provide such a framework, this paper presents four main contributions. The first contribution consists of a general Bayesian Stackelberg game model for security patrolling with execution uncertainty. In this model, execution uncertainty is handled via a Markov decision process (MDP). The second contribution is a detailed study of the problem of computing a Stackelberg equilibrium (SSE) for this game. Computing such SSE in a timely fashion presents significant computational challenges because the defender's strategy space, already exponential in most real-world applications (Jain, Kardes, Kiekintveld, Tambe, & Ordonez, 2010; Conitzer, 2012), only grows in complexity given that it must now address all of the contingencies during execution. To address this shortcoming, we show that when the game's utility functions have a specific separable structure, the defender's strategy space can be compactly represented. By using this structure, we can then reduce the problem to a polynomial-sized optimization problem, which can be solved by existing approaches for solving Bayesian Stackelberg games without execution uncertainty, e.g., DOBSS (Paruchuri, Pearce, Marecki, Tambe, Ordonez, & Kraus., 2008b). However, the randomized patrol schedules that we obtain, are now well-defined MDP-policies, i.e., plans, which take into account contingencies for unexpected events. As we will show in the remainder of this work, such policies can always be generated with a polynomially-sized support. In addition, these policies can be loaded into a smart-phone application carried by patrol units during a shift.

The next two contributions focus on the application of the former approach to generate patrol schedules for fare inspection on the LA Metro. In more detail, the third contribution shows in simulation that, by modeling execution uncertainty as an MDP, we are able to generate policies that overcome the failures of existing SSG algorithms which do not take such uncertainty into account. In addition, results of numerical experiments show that execution uncertainty has a significant impact on the defender's expected utility.

A key question raised for deployed applications of SSGs is the evaluation of their performance in the field. Whereas many different evaluation metrics have been offered, it is difficult to evaluate SSGs approaches and the resource allocation that they generate in actual domains of airport or port security (Tambe, 2011). Fortunately, the MDP policies from our new game model, once loaded onto smartphones—an application we discuss later in this paper—enables us to test the use of SSGs in the field against alternatives. This is a fundamentally new test in the real world to validate not only the new game model, but more generally, algorithmic game theory in the field. Therefore, the fourth contribution is a real-world experiment that aims to evaluate the comprehensive game-theoretic system in the field. Specifically, we ran a 21-day experiment, where we compared schedules generated using our approach against competing schedules comprised of a random scheduler augmented with officers providing real-time knowledge of the current situation. The results provided evidence in support of our MDP-based model—the contingency plans provided by the MDP were actually used with significant frequency in the real world. More importantly, these results showed that game-theoretic schedules led to statistically significant improvements over the competing schedules, despite the fact that the latter were improved with real-time knowledge. These results constitute the first example of head-to-head comparison of SSGs with competing approaches in the field. In fact, they constitute some of the first data obtained about deploying algorithmic game theory in the real-world.

In summary, this paper makes the following contributions:

- We present a novel Bayesian Stackelberg game model that accounts for execution uncertainty in security patrolling using an MDP.

- We study the problem of computing a Stackelberg equilibrium (SSE) for this game. Specifically, we derive conditions under which the game can be represented in a compact form, which can be solved in polynomial time. The resulting strategies are, however, MDP-policies, i.e., plans, which take into account contingencies for unexpected events.

- We present an extensive empirical evaluation whereby we analyze the impact of execution uncertainty on the new game model and on the expected utility for the defender.

- We present a real-world experiment where we compared schedules generated using our approach against competing schedules comprised of a random scheduler. The results showed that game-theoretic schedules outperformed the competing schedules in terms of the number of fare evaders captured. In so doing, they provide evidence about the benefits of deploying algorithmic game theory in the real-world.

The remainder of this paper is organized as follows: Section 2 presents related work on SSGs and how they handle uncertainty. Section 3 discusses the motivating problem of patrolling the LA Metro system and presents the formal model of the problem as a Bayesian Stackelberg game. Section 4 discusses the solution method. Section 5 discusses the way we apply the model defined in Section 4 to the LA Metro problem. Section 6 discusses our evaluation consisting of both simulations and real world experiments and, finally Section 7 concludes and discusses future work.

2. Related Work

Stackelberg security games (SSGs) have gathered significant attention in literature (Basilico et al., 2009a; Dickerson, Simari, Subrahmanian, & Kraus, 2010; Letchford, MacDermed, Conitzer, Parr, & Isbell, 2012; Letchford & Conitzer, 2013; Letchford & Vorobeychik, 2013; Korzhyk, Conitzer, & Parr, 2011a, 2011b). Indeed, as stated earlier, SSGs models and algorithms have been used to build decision aids including ARMOR (Pita et al., 2008), IRIS (Tsai et al., 2009), GUARDS (Pita et al., 2011) and PROTECT (Shieh et al., 2012). Most importantly, two systems, namely TRUSTS (Yin et al., 2012) and RaPtoR (Varakantham, Lau, & Yuan, 2013), have been used to generate schedules for patrolling public transit systems such as the LA Metro and the Singapore Metro system. Unfortunately, all these deployed applications did not take execution uncertainty into account. As a consequence, they are not useful in settings of interest in this paper, such as ones involving patrolling a transportation system where disruptions may occur frequently.

Nonetheless, tackling uncertainty has become one of the principal challenges in SSGs research. In particular, previous work has focused on different types of uncertainties: uncertainty in adversary response due to bounded rationality (Yang, Kiekintveld, Ordonez, Tambe, & John, 2011; Nguyen, Yang, Azaria, Kraus, & Tambe, 2013), uncertainty in adversary surveillance (An, Tambe, Ordonez, Shieh, & Kiekintveld, 2011), uncertainty in adversary capability and uncertainty in defender execution of strategies (Yin et al., 2011; Yin & Tambe, 2012)[2]. With respect to bounded rationality, previous approaches have focused on different models of bounded rationality, such as logit quantal response and subjective utility quantal response (Yang et al., 2011; Nguyen et al., 2013). However, both these frameworks do not address execution uncertainty and, as a consequence, do not address the challenge studied in this work. With respect to adversary surveillance, previous approaches have focused on modeling the fact that in many domains the adversary can only partially observe the defender's mixed strategy (An, Kempe, Kiekintveld, Shieh, Singh, & Tambe, 2012). Similarly, with respect to uncertainty in the adversary's surveillance capability and in the defender's execution of strategy, previous approaches have focused on modeling uncertainty using a Bayesian game (Yin & Tambe, 2012) and on using robust strategy computation, including robust optimization, to provide safe quality guarantees for the obtained defender's strategy (Yin et al., 2011). Unfortunately, as discussed in Section 1, these approaches do not suggest any alternative whenever the defenders on patrol are interrupted. Thus they do not address the challenge in our setting, because they would generate schedules that would become useless anytime a defender is interrupted.

From a game theoretic perspective then, the game model in this paper can be considered as an extensive-form Stackelberg games with chance nodes (Letchford & Conitzer, 2010), or as a special case of a stochastic Stackelberg game where the follower can only choose one action in the initial state and stick to that action in all future states (Letchford et al., 2012). The general cases of both games were shown to be NP-hard. Vorobeychik and Singh provided mixed integer linear programs for finding optimal and approximate Markov stationary strategy in general-sum stochastic Stackelberg games (Vorobeychik & Singh, 2012). However, their approach does not handle multiple adversary types and their MILP formulation lacks the scalability to a large number of states such as the LA Metro problems. Another related line of research is on equilibrium refinement for dynamic games, such as trembling hand perfect equilibrium (Aoyagi, 1996), which considers the possibility that a strategy can be imperfectly executed. However such research is mainly interested in the limit as uncertainty goes to zero, while in our real world settings the probability of imperfect execution really is non-zero.

Other types of SSGs include multi-robot adversarial patrolling games (MAPG). A MAPG is a special restricted type of SSG which considers the problem of multi-robot patrol around a closed area with the existence of an adversary attempting to penetrate into the area (Agmon, Kraus, & Kaminka, 2008a; Agmon, Kaminka, & Kraus, 2011). The penetration requires time and the defender should identify the attacker during her attempt. Most literature about uncertainty in MAPGs studied uncertainty related to the type and

2. Execution uncertainty has also been studied in the context of finding Nash-equilibrium in standard multi-player simultaneous move games (Bowling & Veloso, 2004; Archibald & Shoham, 2011). Despite addressing a similar topic, however, this literature is out of the scope of our work, which is centered on modeling execution uncertainty in SSGs.

the knowledge of the attacker (Agmon, Sadov, Kaminka, & Kraus, 2008b; Basilico, Gatti, Rossi, Ceppi, & Amigoni, 2009b; Basilico, Gatti, & Villa, 2010). Furthermore, it is assumed that if an adversary is detected the robots continue to patrol around the area looking for additional attackers without the need to modify their strategy. Thus, execution uncertainty, as discussed in this work, is not addressed.

One exception, which is closer to our work, is the work by Sless, Agmon, and Kraus (2014) that requires the robots to physically inspect penetration attempts for a given time period. This, as in our work, can have far reaching consequences on the performance of the patrol algorithm. Specifically, it creates vulnerability points along the patrol path that can be taken advantage of by a knowledgeable adversary. In particular, Sless et al. (2014) investigate the problem of coordinated attacks, in which the adversary initiates two attacks in order to maximize its chances of successful penetration, assuming a robot from the team will be sent to examine a penetration attempt. They suggest an algorithm for computing the robots' strategy for handling such coordinated attacks, and show that despite its exponential time complexity, practical run time of the algorithm can be significantly reduced without harming the optimality of the strategy. Unfortunately, whereas this work assumes that the contribution of multiple patrol units covering the same edge is additive, thus enabling to formulate the problem as a linear programming problem, in Sless et al. settings this does not hold because multiple robot covering the same segment contribute the same as a single robot.

Finally, since a significant portion of this work deals with deploying game-theoretic schedules in the field, it is relevant to discuss existing literature that has addressed a similar challenge. As will be discussed in Section 6, we will deploy game-theoretic schedules to deter fare evasion in the LA metro system. In so doing, our work is similar to a number of studies on fare-evasion prevention conducted in the systems of London and Alberta (Clarke, 1993; Weidner, 1996; Clarke, Contre, & Petrossian, 2010). These studies focused on understanding the impact of introducing automatic gates, turn-styles and ticket prices, on the fare evasion rate. In our work, we are interested in a different aspect: understanding how game-theoretic scheduling will affect the performance of the security resources responsible for patrolling a transit system every day.

Given this focus on validating game-theoretic scheduling in the real world, our work shares many ideas with literature on game theory in the field. This line of research has focused on showing equilibrium concepts in the human and animal activities (Ostling, Wang, Tao-yi, Chou, & Camerer, 2011; Brown, Camerer, & Lovallo, 2012). Our work shares their enthusiasm of taking game theory to the field, but fundamentally focuses on *algorithmic deployments* and the impact of such algorithms.

3. Problem Statement

This section discusses, first, the Los Angeles Metro domain, the key domain which is used as a motivation for our work. Second, it presents the Stackelberg game model which we define to formalize the problem.

3.1 Motivating Example: LA Metro System

While our model is quite general for modeling time-sensitive patrols in security domains with execution uncertainty, the study in this paper is substantially motivated by TRUSTS, an application for scheduling fare inspections in the Los Angeles Metro Rail system (Yin et al., 2012). The LA Metro Rail system, similar to other proof-of-payment transit systems worldwide, is a barrier-free transit system where passengers are legally required to purchase tickets before boarding, but are not physically blocked by gates or turnstiles. Instead, security personnel are dynamically deployed throughout the transit system, randomly inspecting passenger tickets. With approximately 300,000 daily riders, the revenue loss due to fare evasion can be significant—this cost has been estimated at $5.6 million (Booz Allen Report, 2007). The Los Angeles Sheriffs Department (LASD) deploys uniformed patrols onboard trains and at stations for fare-checking (and for other purposes such as crime suppression), in order to discourage fare evasion. With limited resources to devote to patrols, it is impossible to cover all locations at all times.

TRUSTS, currently being evaluated at LASD, provides a game-theoretic solution to scheduling randomized patrols for fare evasion deterrence. For a given day, TRUSTS generates one patrol schedule for each fare inspection team according to a pre-computed probability distribution over a large set of possible patrol candidates. A patrol schedule generated is a sequence of fare-check operations, alternating between *in-station* and *on-train* operations. Each operation indicates specifically where and when a patrol unit should check fares. Unfortunately, the security personnel may deviate from the given schedule for a variety of reasons, such as writing citations, felony arrests, handling emergencies, etc. Indeed, in 5 real world trials carried out by the LASD, 4 times the pre-generated schedules got interrupted. Often the entire schedule got abandoned after the interruption if the operations specified afterwards became irrelevant. For example, an officer following a pre-generated schedule had to write a citation to a rider not carrying a valid ticket, preventing her from carrying out the rest of the schedule.

3.2 Formal Model

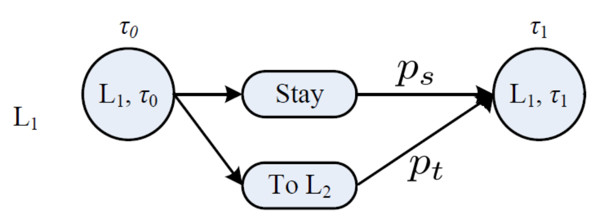


Figure 1: Example of a schedule.

As the first contribution of this work, we present a formal game-theoretic model for patrolling with dynamic execution uncertainty. A *patrolling game with execution uncertainty* is a two-player Bayesian Stackelberg game, between a leader (the defender) and a follower (the adversary). The leader has γ patrol units, and commits to a randomized daily patrol schedule for each unit. A *patrol schedule* consists of a list of commands to be carried out in sequence. Each command is of the form: at time τ, the unit should be at location l, and should execute patrol task a. The patrol action a of the current command, if executed successfully, will take the unit to the location and time of the next command. A graphical representation of a schedule is shown in Figure 1. Each unit faces uncertainty in the execution of each command: delays, or being called to deal with emergencies (possibly at another location). As a result the unit may end up at a location and a time that is different from the intended outcome of the action, and thus the rest of the naive patrol schedule cannot be executed.

We use Markov Decision Processes (MDPs) as a compact representation to model each individual defender unit's patrol actions. We emphasize that these MDPs are not the whole game: they only model the defender's patrol actions with the environment when executing patrols; we will later describe the interaction between the defender and the adversary. Formally, for each defender unit $i \in \{1, \ldots, \gamma\}$ we define an MDP (S_i, A_i, T_i, R_i), where

- S_i is a finite set of states. Each state $s_i \in S_i$ is a tuple (l, τ) of the current location of the unit and the current discretized time. We denote by $l(s_i)$ and $\tau(s_i)$ the location and time of s_i, respectively.
- A_i is a finite set of actions. Let $A_i(s_i) \subseteq A_i$ be the set of actions available at state s_i.
- For each $s_i \in S_i$ and each action $a_i \in A_i(s)$, the *default next state* $n(s_i, a_i) \in S_i$ is the intended next state when executing action a_i at s_i. We call a transition (s_i, a_i, s'_i) a default transition if $s'_i = n(s_i, a_i)$ and a non-default transition otherwise.
- $T_i(s_i, a_i, s'_i)$ is the probability of next state being s'_i if the current state is s_i and the action taken is a_i.
- $R_i(s_i, a_i, s'_i)$ is the immediate reward for the defender from the transition (s_i, a_i, s'_i). For example, being available for emergencies (such as helping a lost child) is an important function of the police, and we can take this into account in our optimization formulation by using R_i to give positive rewards for such events.

We assume that the MDP is acyclic: $T_i(s_i, a_i, s'_i)$ is positive only when $\tau(s'_i) > \tau(s_i)$, i.e., all transitions go forward in time. $S_i^+ \subseteq S_i$ is the subset of states where a patrol could start. A patrol could end at any state. For convenience, we add a dummy source state $s_i^+ \in S_i$ that has actions with deterministic transitions going into each of the states in S_i^+, and analogously a dummy sink state $s_i^- \in S_i$. Thus each patrol of defender i starts at s_i^+ and ends at s_i^-. A patrol execution of i is specified by its *complete trajectory* $t_i = (s_i^+, a_i^+, s_i^1, a_i^1, s_i^2, \ldots, s_i^-)$, which records the sequence of states visited and actions performed. A joint complete trajectory, denoted by $\mathbf{t} = (t_1, \ldots, t_\gamma)$, is a tuple of complete trajectories of all units. Let $\mathcal{X}$ be the finite space of joint complete trajectories.

The immediate rewards R_i are not all the utility received by the defender. The defender also receives rewards from interactions with the adversary. The adversary can be of a set Λ of possible *types* and has a finite set of actions $\mathcal{A}$. The types are drawn from a known distribution, with p_λ the probability of type $\lambda \in \Lambda$. The defender does not know the instantiated type of the adversary, while the adversary does and can condition his decision on his type.

In this general game model, the utilities resulting from defender-adversary interaction could depend arbitrarily on the complete trajectories of the defender units. Formally, for a joint complete trajectory $\mathbf{t}$, the realized adversary type $\lambda \in \Lambda$, and an action of the adversary $\alpha \in \mathcal{A}$, the defender receives utility $u^d(\mathbf{t}, \lambda, \alpha)$, while the adversary receives $u^a(\mathbf{t}, \lambda, \alpha)$.

We are interested in finding the Strong Stackelberg Equilibrium (SSE) of this game, in which the defender commits to a randomized policy which we define next, and the adversary plays a best response to this randomized policy. It is sufficient to consider only pure strategies for the adversary (Conitzer & Sandholm, 2006). Finding one SSE is equivalent to the following optimization problem:

$$\max_{\pi} \sum_{\lambda \in \Lambda} p_\lambda E_{\mathbf{t} \sim \pi}[u^d(\mathbf{t}, \lambda, \alpha_\lambda) + \sum_i R_i(t_i)] \tag{1}$$

$$\text{s.t. } \alpha_\lambda \in \arg\max_{\alpha_\lambda} E_{\mathbf{t} \sim \pi}[u^a(\mathbf{t}, \lambda, \alpha_\lambda)], \forall \lambda \in \Lambda \tag{2}$$

where $R_i(t_i)$ is the total immediate reward from the trajectory t_i, and $E_{\mathbf{t}\sim\pi}[\cdot]$ denotes the expectation over joint complete trajectories induced by defender's randomized policy π.

Whereas MDPs always have Markovian and deterministic optimal policies, in our game the defender's optimal strategy may be non-Markovian because the utilities depend on trajectories, and may be randomized because of interactions with the adversary. We consider two cases: coupled execution and decoupled execution. In coupled execution, patrol units can coordinate with each other; that is, the behavior of unit i at s_i could depend on the earlier joint trajectory of all units. Formally, let $\mathcal{T}_i$ be the set of unit i's *partial trajectories* $(s_i^+, a_i^+, s_i^1, a_i^1, \ldots, s_i')$. A coupled randomized policy is a function $\pi : \prod_i \mathcal{T}_i \times \prod_i A_i \to \mathbb{R}$ that specifies a probability distribution over joint actions of units for each joint partial trajectory. Let $\varphi(\mathbf{t}; \pi) \in \mathbb{R}$ be the probability that joint complete trajectory $\mathbf{t} \in \mathcal{X}$ is instantiated under policy π. In decoupled execution, patrol units do not communicate with each other. Formally, a decoupled randomized policy $\pi = (\pi_1, \ldots, \pi_\gamma)$ where for each unit i, $\pi_i : \mathcal{T}_i \times A_i \to \mathbb{R}$ specifies a probability distribution over i's actions given each partial trajectory of i. Thus a decoupled randomized policy $(\pi_1, \ldots, \pi_\gamma)$ can be thought of as a coupled randomized policy π' where $\pi'(\mathbf{t}, (a_1, \ldots, a_\gamma)) = \prod_i \pi_i(t_i, a_i)$.

Coupled execution potentially yields higher expected utility than decoupled execution. Suppose the defender wants to protect an important target with at least one unit, and unit 1 is assigned that task. Then if she knows unit 1 is dealing with an emergency and unable to reach that target, she can reroute unit 2 to cover the target. However, coordinating among units presents significant logistical and (as we will see in this paper) computational burden.

4. Approach

The defender's optimal strategy may be coupled and non-Markovian, i.e., the policy at s could depend on the entire earlier trajectories of all units rather than the current state s. This makes solving the game computationally difficult—the dimension of the space of mixed strategies is exponential in the number of states.

Nevertheless, in many domains, the utilities have additional structure. There has been extensive research on efficient computation of SSE for massive games with structured utility functions (Tambe, 2011), including for the LA Metro domain (Yin et al., 2012), but these works cannot deal with the type of execution uncertainty studied in this paper. In Section 4.1 we show that under the assumption that the utilities have *separable* structure, it is possible to efficiently compute an SSE of patrolling games with execution uncertainty. In Section 4.2 we discuss generating patrol schedules from solutions described in Section 4.1.

In Section 4.3 we present a procedure to extract an optimal mixed strategy but with a polynomial-sized support. In Section 4.4 we consider a more general case with *partially separable* utilities.

4.1 Efficient Computation via Compact Representation of Strategies

Consider a coupled strategy π. Denote by $x_i(s_i, a_i, s'_i)$ the marginal probability of defender unit i reaching state s_i, executing action a_i, and ending up at next state s'_i. Formally,

$$x_i(s_i, a_i, s'_i) = \sum_{\mathbf{t} \in \mathcal{X}} \varphi(\mathbf{t}; \pi)\theta(t_i, s_i, a_i, s'_i), \tag{3}$$

where the value of the membership function $\theta(t_i, s_i, a_i, s'_i)$ is equal to 1 if trajectory t_i contains transition (s_i, a_i, s'_i) and is equal to 0 otherwise. As defined in Section 3.2, each state is a tuple (l, τ) of the current location of the unit and the current discretized time. Hence, each marginal probability $x_i(s_i, a_i, s'_i)$ takes into account not only the location, but also the time when a specific patrol action is taken. As we will see in the remainder of this section, both time and location affect the expected utility of both players. Hence, this design choice improves the accuracy of the model compared to the real world problem. Let $\mathbf{x} \in \mathbb{R}^M$ be the vector of these marginal probabilities, where $M = \sum_i |S_i|^2|A_i|$. Similarly, let $w_i(s_i, a_i)$ be the marginal probability of unit i reaching s_i and taking action a_i. Let $\mathbf{w} \in \mathbb{R}^{\sum_i |S_i||A_i|}$ be the vector of these marginal probabilities.

Our goal is to compactly represent the SSE problem in terms of $\mathbf{w}$ and $\mathbf{x}$, which have dimensions polynomial in the sizes of the MDPs. We first show that $\mathbf{w}$ and $\mathbf{x}$ satisfy the linear constraints:

$$x_i(s_i, a_i, s'_i) = w_i(s_i, a_i)T_i(s_i, a_i, s'_i), \forall s_i, a_i, s'_i \tag{4}$$

$$\sum_{s'_i, a'_i} x_i(s'_i, a'_i, s_i) = \sum_{a_i} w_i(s_i, a_i), \forall s_i \tag{5}$$

$$\sum_{a_i} w_i(s_i^+, a_i) = \sum_{s'_i, a'_i} x_i(s'_i, a'_i, s_i^-) = 1, \tag{6}$$

$$w_i(s_i, a_i) \geq 0, \forall s_i, a_i \tag{7}$$

Lemma 1. *For all coupled randomized policy π, the resulting marginal probabilities $w_i(s_i, a_i)$ and $x_i(s_i, a_i, s'_i)$ satisfy constraints* (4), (5), (6), (7).

Proof. Constraint (4) holds by the definition of transition probabilities of MDPs. Constraint (5) holds because both *lhs* and *rhs* equal the marginal probability of reaching state s. Constraint (6) holds because by construction, the marginal probability of reaching s_i^+ is 1, and so is the marginal probability of reaching s_i^-. Constraint (7) holds because $w_i(s_i, a_i)$ is a probability. □

Similar formulations for marginal probabilities of MDPs are known (e.g., Filar & Vrieze, 1996). However, unlike in MDPs, in general our utility functions can depend on the defender's complete trajectory and the adversary's type and action, and as a result $\mathbf{w}$ and $\mathbf{x}$ are not sufficient to determine the expected utilities of the game. Thus, in order to make use of this compact representation, we need to make further restrictions to the structure of the utility functions. It turns out that we can formulate expected utilities in terms of **w** and **x** if the game's utilities are *separable*, which intuitively means that given the adversary's strategy, the utilities of both players are sums of contributions from individual units' individual transitions:

Definition 1. *A patrolling game with execution uncertainty as defined in Section 3.2 has* **separable** *utilities if there exist utilities* $U^d_\lambda(s_i, a_i, s'_i, \alpha)$ *and* $U^a_\lambda(s_i, a_i, s'_i, \alpha)$ *for each unit* i*, transition* (s_i, a_i, s'_i)*,* $\lambda \in \Lambda$*,* $\alpha \in \mathcal{A}$*, such that for all* $\mathbf{t} \in \mathcal{X}$*,* $\lambda \in \Lambda$*,* $\alpha \in \mathcal{A}$*, the defender's and the adversary's utilities can be expressed as*

$$u^d(\mathbf{t}, \lambda, \alpha) = \sum_i \sum_{s_i, a_i, s'_i} \theta(t_i, s_i, a_i, s'_i) U^d_\lambda(s_i, a_i, s'_i, \alpha)$$

and

$$u^a(\mathbf{t}, \lambda, \alpha) = \sum_i \sum_{s_i, a_i, s'_i} \theta(t_i, s_i, a_i, s'_i) U^a_\lambda(s_i, a_i, s'_i, \alpha),$$

respectively.

Let $U^d_\lambda, U^a_\lambda \in \mathbb{R}^{M \times |\mathcal{A}|}$ be the corresponding matrices. Then U^d_λ, U^a_λ completely specifies the utility functions u^d and u^a.

Recall that $\theta(t_i, s_i, a_i, s'_i)$ is equal to 1 if trajectory t_i contains transition (s_i, a_i, s'_i) and is equal to 0 otherwise. So the above definition is saying that in a separable game, each player's utility when the trajectory is $\mathbf{t}$ can be decomposed to contributions from each transition of each unit's trajectory t_i. This is a natural extension of the additive reward model of MDPs to the multi-player setting. Separable games can represent common attacker-defender patrolling scenarios, as illustrated in the following example.

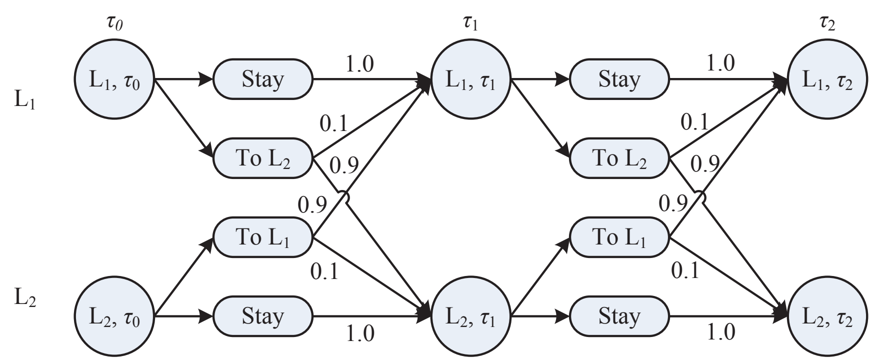


Figure 2: Example game with separable utilities.

Example 1. *Consider a simple example game with one defender unit, whose MDP is illustrated in Figure 2. There are six states, shown as circles in the figure, over two locations* L_1, L_2 *and three time points* τ_0, τ_1, τ_2*. From states at* τ_0 *and* τ_1*, the unit has two actions: to stay at the current location, which always succeeds, and to try to go to* *the other location, which with probability 0.9 succeeds and with probability 0.1 fails (in which case it stays at the current location). There are 12 transitions in total, which is fewer than the number of complete trajectories (18). There is a single type of adversary who chooses one location between L_1 and L_2 and one time point between τ_1 and τ_2 to attack (τ_0 cannot be chosen). If the defender is at that location at that time, the attack fails and both players get zero utility. Otherwise, the attack succeeds, and the adversary gets utility 1 while the defender gets -1. In other words, the attack succeeds if and only if it avoids the defender unit's trajectory. This game has separable utilities: for any transition (s_i, a_i, s'_i) in the MDP, let $U^a_\lambda(s_i, a_i, s'_i, \alpha)$ be 0 if α coincides with s'_i and 1 otherwise. Then each player's utility given a trajectory $\mathbf{t}$ can be expressed as a sum of contributions from transitions, exactly as in Definition 1. For example, the utility expression for the adversary given trajectory $((L_1, \tau_0), \text{"To } L_2\text{"}, (L_1, \tau_1), \text{"To } L_2\text{"}, (L_2, \tau_2))$ is $U^a_\lambda((L_1, \tau_0), \text{"To } L_2\text{"}, (L_1, \tau_1), \alpha) + U^a_\lambda((L_1, \tau_1), \text{"To } L_2\text{"}, (L_2, \tau_2), \alpha)$, which gives the correct utility value for the adversary: 0 if α equals (L_1, τ_1) or (L_2, τ_2) and 1 otherwise.*

It is straightforward to show the following.

Lemma 2. *Consider a game with separable utilities. Suppose $\mathbf{x}$ is the vector of marginal probabilities induced by the defender's randomized policy π. Let $\mathbf{y}_\lambda \in \mathbb{R}^{|\mathcal{A}|}$ be a vector describing the mixed strategy of the adversary of type λ, with $y_\lambda(\alpha)$ denoting the probability of choosing action α. Then the defender's and the adversary's expected utilities from their interactions are $\sum_\lambda p_\lambda \mathbf{x}^T U^d_\lambda \mathbf{y}_\lambda$ and $\sum_\lambda p_\lambda \mathbf{x}^T U^a_\lambda \mathbf{y}_\lambda$, respectively.*

In other words, given the adversary's strategy, the expected utilities of both players are linear in the marginal probabilities $x_i(s_i, a_i, s'_i)$. Lemma 2 also applies when (as in an SSE) the adversary is playing a pure strategy, in which case $\mathbf{y}_\lambda$ is a 0-1 integer vector with $y_\lambda(\alpha) = 1$ if α is the action chosen. We can thus use this compact representation of defender strategies to rewrite the formulation for SSE (1) as a polynomial-sized optimization problem.

$$\max_{\mathbf{w},\mathbf{x},\mathbf{y}} \sum_{\lambda\in\Lambda} p_\lambda \mathbf{x}^T U^d_\lambda \mathbf{y}_\lambda + \sum_{i=1}^{\gamma} \sum_{s_i,a_i,s'_i} x_i(s_i, a_i, s'_i) R_i(s_i, a_i, s'_i) \tag{8}$$

$$\text{s.t. constraints (4), (5), (6), (7)}$$

$$\sum_\alpha y_\lambda(\alpha) = 1, \quad \forall \lambda \in \Lambda \tag{9}$$

$$y_\lambda(\alpha) \in \{0, 1\}, \quad \forall \lambda \in \Lambda,\ \alpha \in \mathcal{A} \tag{10}$$

$$\mathbf{y}_\lambda \in \arg\max_{\mathbf{y}'_\lambda} \mathbf{x}^T U^a_\lambda \mathbf{y}'_\lambda \tag{11}$$

As we will show in Section 4.2, given a solution $\mathbf{w}, \mathbf{x}$ to (8), we can calculate a *decoupled* policy that matches the marginals $\mathbf{w}, \mathbf{x}$. Compared to (1), the optimization problem (8) has exponentially fewer dimensions; in particular the numbers of variables and constraints are polynomial in the sizes of the MDPs. Furthermore, existing methods for solving Bayesian Stackelberg games, such as mixed-integer linear program formulation (Paruchuri, Pearce, Marecki, Tambe, Ordonez, & Kraus, 2008a) or branch-and-bound approach (Yin & Tambe, 2012), can be adapted to solve (8). For example, Paruchuri et al. (2008a) formulated the problem as a mixed-integer quadratic program, then transformed it to an equivalent mixed-integer linear program, which can be solved using standard optimization solvers like CPLEX. While Paruchuri et al. (2008a) assumed that the defender's strategy space is the standard simplex, we can replace the simplex constraints with the flow constraints (4), (5), (6), (7), and apply the same techniques described by Paruchuri et al. (2008a) to derive a mixed-integer linear program.

For the special case of $U^d_\lambda + U^a_\lambda = \mathbf{0}$ for all λ, i.e., when the interaction between defender and adversary is zero-sum, the SSE problem can be formulated as the following linear program (LP):

$$\max_{\mathbf{w},\mathbf{x},\mathbf{u}} \sum_{\lambda\in\Lambda} p_\lambda u_\lambda + \sum_i \sum_{s_i,a_i,s'_i} x_i(s_i,a_i,s'_i)R_i(s_i,a_i,s'_i) \tag{12}$$

$$\text{s.t.}\quad \text{constraints (4), (5), (6), (7)}$$

$$u_\lambda \leq \mathbf{x}^T U^d_\lambda \mathbf{e}_\alpha, \quad \forall\lambda\in\Lambda,\ \alpha\in\mathcal{A}, \tag{13}$$

where $\mathbf{e}_\alpha$ is the basis vector corresponding to adversary action α. This LP is similar to the maximin LP for a zero-sum game with the utilities given by U^d_λ and U^a_λ, except that an additional term $\sum_i \sum_{s_i,a_i,s'_i} x_i(s_i,a_i,s'_i)R_i(s_i,a_i,s'_i)$ representing defender's expected utilities from immediate rewards is added to the objective. One potential issue arises: because of the extra defender utilities from immediate rewards, the entire game is no longer zero-sum. Is it still valid to use the above maximin LP formulation? It turns out that the LP is indeed valid, as the immediate rewards do not depend on the adversary's strategy.

Proposition 1. *If the game has separable utilities and $U^d_\lambda + U^a_\lambda = 0$ for all λ, then a solution of the LP* (12) *is an SSE.*

Proof. We can transform this game to an equivalent zero-sum Bayesian game whose LP formulation is equivalent to (12). Specifically, given the non-zero-sum Bayesian game Γ specified above, consider the Bayesian game Γ' with the following "meta" type distribution for the second player: for all $\lambda\in\Lambda$ of Γ there is a corresponding type $\lambda'\in\Lambda'$ in Γ', with probability $p_{\lambda'} = 0.5p_\lambda$, with the familiar utility functions; and there is a special type $\phi\in\Lambda'$ with probability $p_\phi = 0.5$, whose action does not affect either player's utility. Specifically the utilities under the special type ϕ are $u^d(\mathbf{t},\phi,\alpha) = \sum_i \sum_{s_i,a_i,s'_i} \theta(t_i,s_i,a_i,s'_i)R_i(s_i,a_i,s'_i)$ and $u^a(\mathbf{t},\phi,\alpha) = -\sum_i \sum_{s_i,a_i,s'_i} \theta(t_i,s_i,a_i,s'_i)R_i(s_i,a_i,s'_i)$. The resulting game Γ' is zero-sum, with the defender's utility exactly half the objective of (12). Since for zero-sum games maximin strategies and SSE coincide, a solution of the LP (12) is an optimal SSE marginal vector for the defender of Γ'. On the other hand, if we compare the induced normal forms of Γ and Γ', the only difference is that for the adversary the utility $-0.5\sum_{e\in E^*} U_e x_e$ is added, which does not depend on the adversary's strategy. Therefore Γ and Γ' have the same set of SSE, which implies that a solution of the LP is an SSE of Γ. $\square$

4.2 Generating Patrol Schedules

The solution of (8) does not yet provide a complete specification of what to do. We ultimately want an explicit procedure for generating the patrol schedules. We define a *Markov strategy* π to be a decoupled strategy $(\pi_1,\dots,\pi_\gamma)$, $\pi_i : S_i \times A_i \to \mathbb{R}$, where the distribution over next actions depends only on the current state. Proposition 2 below shows that given $\mathbf{w}, \mathbf{x}$, there is a simple procedure to calculate a Markov strategy that matches the marginal probabilities. This implies that if $\mathbf{w}, \mathbf{x}$ is the optimal solution of (8), then the corresponding Markov strategy π achieves the same expected utility. We have thus shown that for games with separable utilities it is sufficient to consider Markov strategies.

Proposition 2. *Given* $\mathbf{w}, \mathbf{x}$ *satisfying constraints* (4) *to* (7)*, construct a Markov strategy* π *as follows: for each* $s_i \in S_i$*, for each* $a_i \in A_i(s_i)$*,* $\pi_i(s_i, a_i) = \frac{w_i(s_i,a_i)}{\sum_{a'_i} w_i(s_i,a'_i)}$*. If* $\sum_{a'_i} w_i(s_i, a'_i) = 0$ *we set* $\pi_i(s_i, \cdot)$ *to be an arbitrary distribution. Suppose the defender plays* π*, then for all unit* i *and transition* (s_i, a_i, s'_i)*, the probability that* (s_i, a_i, s'_i) *is reached by* i *equals* $x_i(s_i, a_i, s'_i)$*.*

Proof. Such a Markov strategy π induces a Markov chain over the states S_i for each unit i. We claim that the resulting marginal probability vector of this Markov chain matches $\mathbf{x}$. We can show this by induction from the starting state s_i^+ to successor states. The marginal probability $\Pr(s_i^+, a_i)$ of reaching state s_i^+ and taking action a_i is equal to $\pi_i(s_i^+, a_i)$, since s_1^+ is always reached. But $\pi_i(s_i^+, a_i) = \frac{w_i(s_i^+,a_i)}{\sum_{a'_i} w_i(s_i^+,a'_i)} = w_i(s_i^+, a_i)$ by constraint (6), So the marginals are matched at s_i^+. For the inductive step, the marginal probability $\Pr(s_i, a_i)$ of reaching s_i and taking action a_i is equal to $\Pr(s_i)\pi_i(s_i, a_i)$, where $\Pr(s_i)$ is the probability of reaching s_i. By the induction hypothesis, $\Pr(s_i)$ can be computed from the marginals $\mathbf{x}, \mathbf{w}$ on the previous states, as $\Pr(s_i) = \sum_{s'_i,a'_i} x_i(s'_i, a'_i, s_i)$. Then

$$\Pr(s_i, a_i) = \pi_i(s_i, a_i) \sum_{s'_i,a'_i} x_i(s'_i, a'_i, s_i) = \frac{w_i(s_i, a_i)}{\sum_{a'_i} w_i(s_i, a'_i)} \sum_{s'_i,a'_i} x_i(s'_i, a'_i, s_i) = w_i(s_i, a_i)$$

by constraint (5). Thus the marginals are matched at all states. □

In practice, directly implementing a Markov strategy π requires each unit i to draw an action according to the probability distribution $\pi_i(s_i, \cdot)$ at each state s_i. This is possible when each unit can consult a random-number generator, or can communicate with a central command. However, in certain domains such requirement on computation or communication at each time step places additional logistical burden on the patrol unit. To avoid unnecessary computation or communication at every time step, it is desirable to let each unit execute a deterministic schedule (i.e., a pure strategy). To guarantee the optimal expected utility, we want the deterministic schedule to be drawn from a distribution that has the same marginals as the optimal solution of (8). We say a procedure that generates patrols is *correct* if it has this property. With no execution uncertainty, a pure strategy can be specified by the complete trajectory for each unit. However, this no longer works in the case with execution uncertainty, as interruptions will lead to states outside the trajectory.

We thus begin by defining a Markov pure strategy, which specifies a deterministic choice at each state.

Definition 2. *A* **Markov pure strategy q** *is a tuple* $(q_1, \ldots, q_\gamma)$ *where for each unit* i*,* $q_i : S_i \to A_i$*.*

We note that the set of Markov pure strategies is a subset of all pure strategies, which can generally condition their choices on earlier histories in addition to the current states. Nevertheless, we will show that Markov pure strategies are useful as part of an efficient procedure for sampling from the solution of (8).

Given a Markov strategy π, we sample a Markov pure strategy $\mathbf{q}$ as follows:

Procedure 1. *Given π, for each unit i and state $s_i \in S_i$, sample an action a_i according to π_i, and set $q_i(s_i)$ to be a_i. Output the Markov pure strategy $\mathbf{q}$.*

This procedure is correct since each state in i's MDP is visited at most once and thus q_i exactly simulates a walk from s_i^+ on the Markov chain induced by π_i.

To directly implement a Markov pure strategy, the unit needs to either remember the entire mapping $\mathbf{q}$ or to receive the action from a central command at each time step. An alternative scheme that requires a small amount of storage and a minimal amount of communication is the following: the central command sends the unit a trajectory assuming perfect execution, and only after a non-default transition happened does the unit communicates with the central command to get a new trajectory starting from the current state. Formally, given $s_i \in S_i$ and q_i, we define the *optimistic* trajectory from s_i induced by q_i to be $(s_i, q_i(s_i), n(s_i, q_i(s_i)), \dots s^-)$, i.e, the trajectory assuming it always reaches its default next state. Given a Markov pure strategy $\mathbf{q}$, the following procedure exactly simulates $\mathbf{q}$:

Procedure 2. *For each unit i: (i) central command gives unit i the optimistic trajectory from s^+ induced by q_i; (ii) unit i follows the trajectory until the terminal state s^- is reached or some unexpected event happens and takes i to state s_i'; (iii) in the latter case, central command sends unit i the new optimistic trajectory from s_i' induced by q_i and repeat from step (ii).*

4.3 Extracting a Mixed Strategy with Small Support

So far the procedures for generating patrol schedules we described are different implementations of the Markov strategy π from Proposition 2. This corresponds to a mixed strategy over the set of Markov pure strategies: each Markov pure strategy $\mathbf{q}$ is played with probability equal to the probability that it is sampled by the sampling procedure. The support of this mixed strategy, i.e., the set of pure strategies with non-zero probability, is in general an exponential-sized set.

In practice, it is sometimes desirable to have a mixed strategy with polynomial-sized support. For example, the security agency may need to carry out training exercises for each of the pure strategies in the support (e.g., Fang et al., 2013). In such cases, we would like a polynomial-support mixed strategy that achieves the same expected utility as the optimal solution of (8). The following proposition shows that there always exists a polynomial-support mixed strategy that matches the given marginals $\mathbf{w}, \mathbf{x}$, and therefore achieves the optimal expected utility when $\mathbf{w}, \mathbf{x}$ are the solution of (8).

Proposition 3. *Given $\mathbf{w}, \mathbf{x}$ satisfying constraints* (4) *to* (7)*, there exists a mixed strategy with polynomial-sized support that matches the marginals.*

Proof. Since $(\mathbf{w}, \mathbf{x})$ is in the polytope defined by constraints (4) to (7), and extreme points of the polytope correspond to pure strategies, Caratheodory's Theorem[3] implies that $(\mathbf{w}, \mathbf{x})$

3. See Chapter 3 from (Gruber, 2007)

can be written as a convex combination of at most $dim(\mathbf{w})+dim(\mathbf{x})+1 = O(\sum_i |S_i|^2 \times |A_i|)$ pure strategies. $\square$

However, the above proof is not constructive. Grotschel, Lovasz, and Schrijver (1981) provided a polynomial-time procedure that given a point inside a polytope, decomposes it into a convex combination of a polynomial number of the extreme points of the polytope; however their method requires the application of ellipsoid method, which tends to be slow in practice. In this section, we provide an efficient procedure for generating a mixed strategy with polynomial-sized support that matches the marginals $\mathbf{w}, \mathbf{x}$.

Procedure 3. *Given* $(\mathbf{w}, \mathbf{x})$*, an optimal solution of* (8)*, we initialize the mixed strategy to have an empty support set.*

1. *Compute the Markov strategy* π *as* $\pi_i(s_i, a_i) = \frac{w_i(s_i,a_i)}{\sum_{a'_i} w_i(s_i,a'_i)}$*. If* $\sum_{a'_i} w_i(s_i, a'_i) = 0$ *we set* $\pi_i(s_i, \cdot)$ *to be an arbitrary distribution.*

2. *Select a Markov pure strategy* $\mathbf{q}$ *that is played with positive probability in* π*. One possible way to construct* $\mathbf{q}$ *is as follows: at each state* s_i*, set* $\mathbf{q}_i(s_i)$ *to be an action* a_i *that is played with positive probability at* s_i *in* π*.*

3. *Compute the marginals* $\mathbf{w^q}, \mathbf{x^q}$ *corresponding to pure strategy* $\mathbf{q}$*. This can be done from* s_i^+ *to successor states.*

4. *Let* γ_q *be the maximum* γ *such that* $\mathbf{w} - \gamma \mathbf{w^q} \geq 0$ *and* $\mathbf{x} - \gamma \mathbf{x^q} \geq 0$*.*

5. *Set* $\mathbf{w} := \mathbf{w} - \gamma_q \mathbf{w^q}$*,* $\mathbf{x} := \mathbf{x} - \gamma_q \mathbf{x^q}$*. Add pure strategy* $\mathbf{q}$ *to the support, played with probability* γ_q*.*

6. *If* $\mathbf{w} = 0$*, Stop. Otherwise, go to Step 1.*

Proposition 4. *Procedure 3 outputs in polynomial time a mixed strategy with polynomial-sized support, matching the marginals.*

Proof. Since $\mathbf{w^q}, \mathbf{x^q}$ are valid marginal vectors satisfying constraints (4) to (7), the updated marginals $(\mathbf{w}, \mathbf{x})$ after Step 5 still satisfies the flow conservation constraints (4), (5), and the nonnegativity constraint (7), while the total flow (corresponding to (6)) is reduced by γ_q. This implies that the steps of the procedure are well-defined; furthermore once the procedure stops, the output γ's sum to 1, and the corresponding mixed strategy matches the marginals $\mathbf{w^q}, \mathbf{x^q}$.

It remains to show that the procedure stops after a polynomial number of iterations. To see this, note that by construction, each execution of Step 5 will reduce at least one component of $(\mathbf{w}, \mathbf{x})$ to zero. This is because otherwise we could increase γ_q, contradicting Step 4. Therefore the procedure stops after at most $dim(\mathbf{w}) + dim(\mathbf{x}) = O(\sum_i |S_i|^2 \times |A_i|)$ iterations; and since each iteration adds one pure strategy to the support, the resulting mixed strategy has a polynomial-sized support. $\square$

4.4 Coupled Execution: Cartesian Product MDP

Without the assumption of separable utilities, it is no longer sufficient to consider decoupled Markov strategies of individual units' MDPs. We create a new MDP that captures the joint execution of patrols by all units. For simplicity of exposition we look at the case with two defender units. Then a state in the new MDP corresponds to the tuple (location of unit 1, location of unit 2, time). An action in the new MDP corresponds to a tuple (action of unit 1, action of unit 2). Formally, if unit 1 has an action a_1 at state $s_1 = (l_1, \tau)$ that takes her to $s'_1 = (l'_1, \tau')$ with probability $T_1(s_1, a_1, s'_1)$, and unit 2 has an action a_2 at state $s_2 = (l_2, \tau)$ that takes her to $s'_2 = (l'_2, \tau')$ with probability $T_2(s_2, a_2, s'_2)$, we create in the new MDP an action $a_\times = (a_1, a_2)$ from state $s_\times = (l_1, l_2, \tau)$ that transitions to $s'_\times = (l'_1, l'_2, \tau')$ with probability $T_\times(s_\times, a_\times, s'_\times) = T_1(s_1, a_1, s'_1)T_2(s_2, a_2, s'_2)$. The immediate rewards $R_\times$ of the MDP are defined analogously. We call the resulting MDP $(S_\times, A_\times, T_\times, R_\times)$ the Cartesian Product MDP.

An issue arises when at state $s_\times$ the individual units have transitions of different time durations. For example, unit 1 rides a train that takes 2 time steps to reach the next station while unit 2 stays at a station for 1 time step. During these intermediate time steps only unit 2 has a "free choice". How do we model this on the Cartesian Product MDP? One approach is to create new states for the intermediate time steps. For example, suppose at location L_A at time 1 a non-default transition takes unit 1 to location L_A at time 3. We modify unit 1's MDP so that this transition ends at a new state $(L^1_A, 2) \in S_1$, where L^1_A is a "special" location specifying that the unit will become alive again at location L_A in one more time step. There is only one action from $(L^1_A, 2)$, with only one possible next state $(L_A, 3)$. Once we have modified the individual units' MDPs so that all transitions take exactly one time step, we can create the Cartesian Product MDP as described in the previous paragraph.

Like the units' MDPs, the Cartesian Product MDP is also acyclic. Therefore we can analogously define marginal probabilities $w_\times(s_\times, a_\times)$ and $x_\times(s_\times, a_\times, s'_\times)$ on the Cartesian Product MDP. Let $\mathbf{w}_\times \in \mathbb{R}^{|S_\times||A_\times|}$ and $\mathbf{x}_\times \in \mathbb{R}^{|S_\times|^2|A_\times|}$ be the corresponding vectors. Utilities generally cannot be expressed in terms of $\mathbf{w}_\times$ and $\mathbf{x}_\times$. We consider a special case in which utilities are *partially separable*:

Definition 3. *A patrolling game with execution uncertainty has* **partially separable** *utilities if there exist $U^d_\lambda(s_\times, a_\times, s'_\times, \alpha)$ and $U^a_\lambda(s_\times, a_\times, s'_\times, \alpha)$ for each transition $(s_\times, a_\times, s'_\times)$, $\lambda \in \Lambda$, $\alpha \in \mathcal{A}$, such that for all $\mathbf{t} \in \mathcal{X}$, $\lambda \in \Lambda$, $\alpha \in \mathcal{A}$, the defender's and the adversary's utilities can be expressed as $u^d(\mathbf{t}, \lambda, \alpha) = \sum_{s_\times, a_\times, s'_\times} \theta_\times(\mathbf{t}, s_\times, a_\times, s'_\times)U^d_\lambda(s_\times, a_\times, s'_\times, \alpha)$ and $u^a(\mathbf{t}, \lambda, \alpha) = \sum_{s_\times, a_\times, s'_\times} \theta_\times(\mathbf{t}, s_\times, a_\times, s'_\times)U^a_\lambda(s_\times, a_\times, s'_\times, \alpha)$, respectively.*

Partially separable utilities is a weaker condition than separable utilities, as now the expected utilities may not be sums of contributions from individual units. When utilities are partially separable, we can express expected utilities in terms of $\mathbf{w}_\times$ and $\mathbf{x}_\times$ and find an SSE by solving an optimization problem analogous to (8). From the optimal $\mathbf{w}^*_\times$, we can get a Markov strategy $\pi^*_\times(s_\times, a_\times) = \frac{w^*_\times(s_\times, a_\times)}{\sum_{a'_\times} w^*_\times(s, a'_\times)}$, which is provably the optimal coupled strategy.

This approach cannot scale up to a large number of defender units, as the size of $S_\times$ and $A_\times$ grow exponentially in the number of units. In particular the dimension of the Markov policy $\pi_\times$ is already exponential in the number of units. To overcome this we will need a more compact representation of defender strategies. One approach is to use decoupled strategies. The resulting defender strategy model resembles a transition independent DEC-MDP (Becker, Zilberstein, Lesser, & Goldman, 2004). However, due to strategic interaction against adversaries, existing methods for DEC-MDP—which compute pure strategies—cannot be directly applied. Efficient computation in these settings remains an open problem.

5. Case Study on the LA Metro System

This section discusses how we apply the model presented in Section 3.2 and the approach presented in Section 4 to the problem of patrolling the LA Metro system for fare evasion. In particular, Section 5.1 discusses how the game model is adapted to the LA metro patrolling problem and Section 5.2 discusses how we use this framework to build a scheduling system whereby schedules are generated by a central planner and are visualized by a smartphone application running on android phones.

5.1 Application to LA Metro Domain

We now explain how our proposed techniques can be used in the LA Metro domain. This involves defining a number of parameters specific to the LA Metro domain which will be initialized for our real-world experiment in Section 6. In addition, as we will see, although the utilities in this domain are not separable, we are able to provide an upper bound to the defender utilities by separable utilities, allowing efficient computation.

Similar to the work by Yin et al. (2012), a state here comprises the current station and time of a unit, as well as necessary history information such as starting time[4]. At any state, a unit may stay at her current station to conduct an *in-station* operation for some time or she can ride a train to conduct an *on-train* operation when her current time coincides with the train schedule. Due to execution uncertainty, a unit may end up at a state other than the intended outcome of the action. For ease of analysis, we assume throughout the rest of this paper a single type of unexpected event which delays a patrol unit for some time beyond the intended execution time. Specifically, we assume for any fare check operation taken, there is a probability η that the operation will be delayed, i.e., staying at the same station (for *in-station* operations) or on the same train (for *on-train* operations) involuntarily for some time. Furthermore, we assume that units will be involved with events unrelated to fare enforcement and thus will not check fares during any delayed period of an operation. Intuitively, a higher chance of delay leads to less time spent on fare inspection. As we will see in Section 6, initializing this probability for conducting real-world experiments required adopting a robust approach based on cross-validation (Kohavi, 1995; Jaulmes, Pineau, & Precup, 2007) to address the uncertainty related to the duration of a delay.

The adversary faced here are the riders in the system. Specifically, we model the most common type of riders which use the metro line every day to go to their work and come back home. Each of them takes a fixed route: it starts at a specific station A (at a specific time) and ends at a new station B (at a new time). Since there exists multiple stations and time units, there also exist multiple routes to considered. To address this issue, we define

4. Interested readers are encouraged to read the work by Yin et al. (2012) for more details

multiple riders each representing a specific route. A rider observes the likelihood of being checked and makes a binary decision between buying and not buying the ticket. If the rider of type λ buys the ticket, he pays a fixed ticket price ρ_λ. Otherwise, he rides the train for free but risks the chance of being caught and paying a fine of $\varrho_\lambda > \rho_\lambda$. The LASD's objective is set to maximize the overall revenue of the whole system including ticket sales and fine collected, essentially forming a zero-sum game. This revenue depends on the number of resources available to the LASD, i.e., the ones that can be deployed for each patrol[5]. Given a number of available officers then, the costs associated with their deployment (e.g., number of officers and working hours) can be incorporated into the objective function by defining negative rewards in Equation 8.

Since the fare check operation performed is determined by the actual transition rather than the action taken, we define the effectiveness of a transition (s, a, s') against a rider type λ, $f_\lambda(s, a, s')$, as the percentage of riders of type λ checked by transition (s, a, s'). Following the same argument by Yin et al. (2012), we assume the probability that a joint complete trajectory $\mathbf{t}$ detects evader λ as the sum of f_λ over all transitions in $\mathbf{t} = (t_1, \ldots, t_\gamma)$ capped at one:

$$\Pr(\mathbf{t}, \lambda) = \min\{\sum_{i=1}^{\gamma} \sum_{s_i, a_i, s'_i} f_\lambda(s_i, a_i, s'_i)\theta(t_i, s_i, a_i, s'_i), 1\}. \tag{14}$$

For type λ and joint trajectory $\mathbf{t}$, the LASD receives ρ_λ if the rider buys the ticket and $\varrho_\lambda \cdot \Pr(\mathbf{t}, \lambda)$ otherwise. The utilities in this domain are indeed not separable — even though multiple units (or even a single unit) may detect a fare evader multiple times, the evader can only be fined once. As a result, neither players' utilities can be computed directly using marginal probabilities $\mathbf{x}$ and $\mathbf{w}$. Instead, we provide an upper bound to the defender utility by overestimating the detection probability as the following:

$$\widehat{\Pr(\mathbf{t}, \lambda)} = \sum_{i=1}^{\gamma} \sum_{s_i, a_i, s'_i} f_\lambda(s_i, a_i, s'_i)\theta(t_i, s_i, a_i, s'_i) \geq \Pr(\mathbf{t}, \lambda). \tag{15}$$

Then the defender utility if the rider does not buy the ticket is upper-bounded by $\varrho_\lambda \cdot \widehat{\Pr(\mathbf{t}, \lambda)}$, which is separable. Given a marginal vector $\mathbf{x}$, the detection probability is then upper-bounded by

$$\widehat{\Pr(\mathbf{x}, \lambda)} = \sum_{i=1}^{\gamma} \sum_{s_i, a_i, s'_i} f_\lambda(s_i, a_i, s_i) x_i(s_i, a_i, s'_i). \tag{16}$$

Equation (16) leads to the following upper bound LP for the LA Metro problem:

$$\max_{\mathbf{x}, \mathbf{w}, \mathbf{u}} \quad \sum_{\lambda \in \Lambda} p_\lambda u_\lambda + \sum_{i=1}^{\gamma} \sum_{s_i, a_i, s'_i} R_i(s_i, a_i, s'_i) \tag{17}$$

$$\text{s.t.} \quad \text{constraints (4), (5), (6), (7)}$$

$$u_\lambda \leq \min\{\rho_\lambda,\ \varrho_\lambda \cdot \widehat{\Pr(\mathbf{x}, \lambda)}\}, \text{ for all } \lambda \in \Lambda \tag{18}$$

5. The number of resources required to maximize the revenue for the LASD is equal to the number of number of edges of the MDP defined in Section 4. Unfortunately, in the real world the available resources are much less than this number. Therefore, the idea is to maximize the revenue for the LASD given the resources available.

We prove the claims above by the following two propositions.

Proposition 5. $\widehat{\Pr(\mathbf{x}, \lambda)}$ *is an upper bound of the true detection probability of any coupled strategy with marginals* $\mathbf{x}$.

Proof. Consider a coupled strategy π. Recall that $\varphi(\mathbf{t}; \pi) \in \mathbb{R}$ is the probability that joint trajectory $\mathbf{t} \in \mathcal{X}$ is instantiated. For rider type λ, the true detection probability is $\Pr(\pi, \lambda) = \sum_{\mathbf{t} \in \mathcal{X}} \varphi(\mathbf{t}; \pi) \Pr(\mathbf{t}, \lambda)$. Applying Equations (14) and (3) we have,

$$\begin{aligned} \Pr(\pi, \lambda) &\leq \sum_{\mathbf{t} \in \mathcal{X}} \varphi(\mathbf{t}; \pi) \sum_{i=1}^{\gamma} \sum_{s_i, a_i, s'_i} f_\lambda(s_i, a_i, s'_i) \theta(t_i, s_i, a_i, s'_i) \\ &= \sum_{i=1}^{\gamma} \sum_{s_i, a_i, s'_i} f_\lambda(s_i, a_i, s'_i) \sum_{\mathbf{t} \in \mathcal{X}} \varphi(\mathbf{t}; \pi) \theta(t_i, s_i, a_i, s'_i) \\ &= \sum_{i=1}^{\gamma} \sum_{s_i, a_i, s'_i} f_\lambda(s_i, a_i, s'_i) x_i(s_i, a_i, s'_i) = \widehat{\Pr(\mathbf{x}, \lambda)}. \quad \square \end{aligned}$$

$\square$

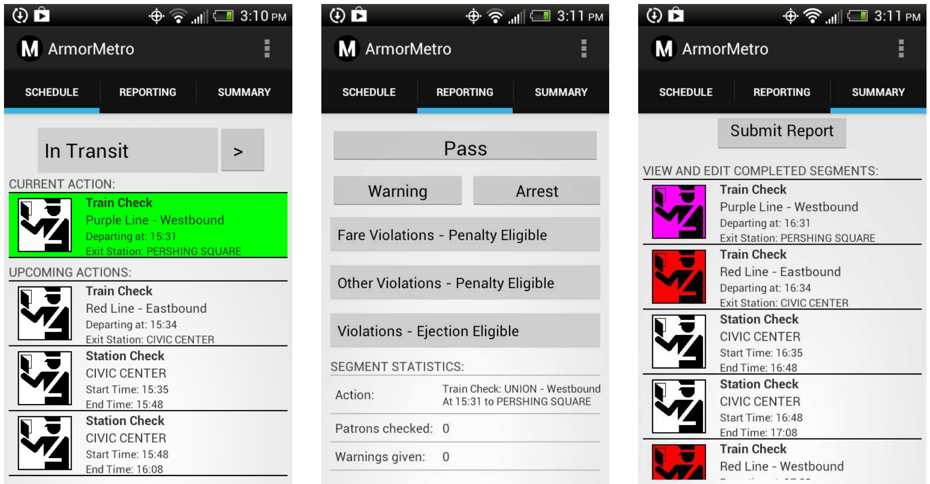
Proposition 6. *LP* (17) *provides an upper bound of the optimal coupled strategy.*

Sketch. Let $\mathbf{x}^*$ and $\mathbf{w}^*$ be the marginal coverage and u^*_λ be the value of the patroller against rider type λ in the optimal coupled strategy π^*. It suffices to show that ($\mathbf{x}^*$, $\mathbf{w}^*$, $\mathbf{u}^*$) is a feasible point of the LP. From Lemma 1, we already know $\mathbf{x}^*$ and $\mathbf{w}^*$ must satisfy constraints (4) to (7). Furthermore, we have $u^*_\lambda \leq \rho_\lambda$ since the rider pays at most the ticket price. Finally, $u^*_\lambda \leq \varrho_\lambda \cdot \widehat{\Pr(\mathbf{x}, \lambda)}$ since $\widehat{\Pr(\mathbf{x}, \lambda)}$ is an overestimate of the true detection probability. $\square$

Intuitively, LP (17) relaxes the utility functions by allowing an evader to be fined multiple times during a single trip. The relaxed utilities are separable and thus the relaxed problem can be efficiently solved. Since the solution returned $\mathbf{x}^*$ and $\mathbf{w}^*$ satisfy constraints (4) to (7), we can construct a Markov strategy from $\mathbf{w}^*$ as described in Section 4.2. The Markov strategy provides an approximate solution to the original problem, whose actual value can be evaluated using Monte Carlo simulation. This strategy is also sampled to produce a patrol schedule which is then uploaded on the smartphone application which is discussed next.

5.2 The Smartphone Application

The smartphone app is a software agent carried by each patrol officer that visualizes the patrol schedules generates using the approach discussed in the previous section. Shown in Figure 3, the app provides three principal features: (i) a patrol schedule for the current shift; (ii) a system for reporting passenger violations and (iii) a shift statistics summary report. At the beginning of the shift, a patrol schedule is loaded into the app either by hand or using a database. The app then displays a schedule of the current and upcoming patrol actions in the schedule view, shown in Figure 3(a). Implementing recovery from



(a) Schedule View (b) Reporting View (c) Summary View

Figure 3: The smartphone' user interface

real-world unexpected events that interrupt the officer's schedule, the schedule view also allows the officer to manually set their current location, triggering the app to select a new patrol schedule based on the officer's current location and time. To do so, a number of patrol schedules are sampled from the Markov strategy (see Section 4.2) and are uploaded in the app before deployment. In the remainder of this work, we will refer to this action as requesting an "update". The app also allows patrol officers to record passenger violations, such as fare evasion, for the current patrol action using Reporting View, shown in Figure 3(b). Officers can also view and edit the passenger violations reported for past actions in Summary View, shown in Figure 3(c). Upon shift completion, the officer can also use Summary View to submit the app-generated shift statistics summary report, including all unexpected events and violations reported throughout the shift, to a database.

This app presents two key advantages for security agencies. First, it allows for the collection of patrol data, which could then be used to analyze the behavior of adversaries such as fare evaders and criminals. Second, this app-collected data could also benefit transit system security departments that manually record violations data or conduct their own analysis on patrol strategy performance.

6. Evaluation

This section describes our experiments. In Section 6.1, we describe the datasets and how we instantiated some of the model parameters for our experiments. Section 6.2, discusses our simulations in which we studied the key properties of the approach discussed in Section 5.1. Finally, Section 6.3 discusses the real-world experiment where we ran a head-to-head comparison between our approach and a uniform random approach, the automated approach that most security agencies use when not using a game-theoretic approach (Tambe, 2011).

6.1 Data Sets

The experiments presented in this work, either in simulation or in the real-world, are based on four data sets, each based on a different Los Angeles Metro Rail line: Red (including Purple), Blue, Gold, and Green. In these data sets, the train schedules were obtained from `http://www.metro.net` and the ridership distributions for each line were estimated from hourly boarding and alighting counts provided by the LASD. We allowed any *on-train* operation, i.e., train checks could be between two or more stations and defined *in-station* operations, i.e., station-checks, to be fixed intervals between 10 and 20 minutes. As recommended by the LASD, this duration is the one that security officers typically adopt to conduct fare inspections at a train station. The effectiveness of each fare check operation was adjusted based on the volume of riders during the period with an assumption that a unit would check three riders per minute. The ticket fare was set to \$1.5 (the actual current value), while the fine was set to \$100 (fare evaders in Los Angeles can be fined \$200, however, they also may be issed warnings). A rider can always pay the ticket price for \$1.5 and will only evade the ticket when the expected fine is lower. The immediate rewards R_i are all set to zero. Table 1 summarizes the detailed statistics for the four Metro lines.

Line	Stops	Trains	Daily Riders	Types
Red	16	433	149991.5	26033
Blue	22	287	76906.2	46630
Gold	19	280	30940.0	41910
Green	14	217	38442.6	19559

Table 1: Statistics of Los Angeles Metro lines.

6.2 Simulations

The simulations are aimed to analyze the performance of the Markov strategies calculated solving the LP defined in Section 5.1. More specifically, we aim to analyze the key features of our approach by systematically manipulating those parameters that are more likely to vary significantly in the real world and, as consequence, affect the revenue for the defender and the rate of fare evaders captured. These parameters include the delay length, the train lines, the levels of execution uncertainty and the number of available resources. We also tested the runtime of our LP algorithm to verify whether it was capable of generating an output in a timely manner. Most result are presented for the Red line only. Results for the Blue, the Gold and the Green line are presented in the Appendix A.

In all such simulations, we found that the Markov strategy was close to optimal with revenue always above 99% of the LP upper bound. Figure 4 shows such a result, assuming 6 resources patrolling the Red line for 3 hours and varying uncertainty probability (from 0% to 25%). Similar results were found for the Blue, Green and Gold lines and are reported in Figure 14 in the Appendix A. This result indicated that the relaxed detection probability given in Equation (16) provided a good estimation of the true probability, implying that riders were unlikely to be checked more than once by joint execution trajectories produced by the Markov strategy in all of our data sets. For this reason, in the remainder of this

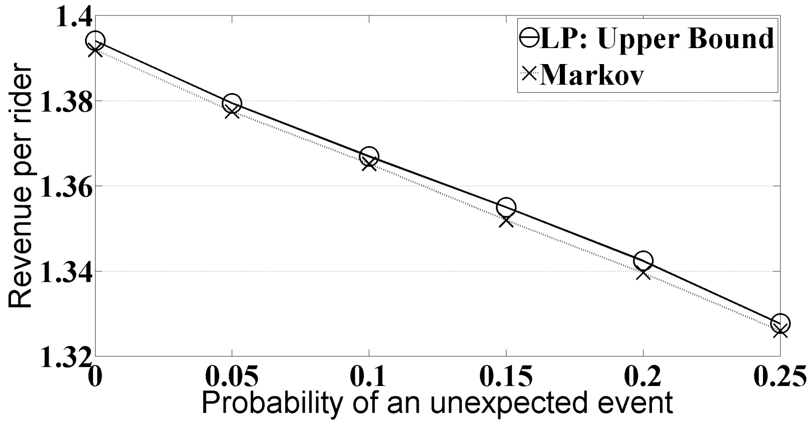


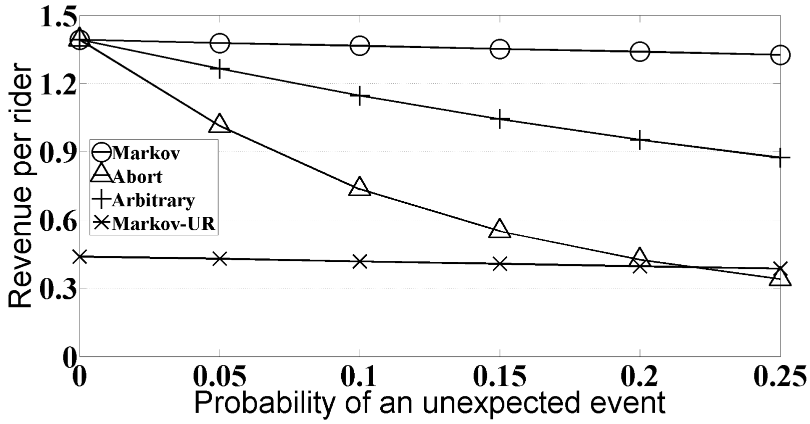
Figure 4: the Markov Strategy (Equation (17)) vs. the true LP (Equation (8)).

section we will report values of the Markov strategy without mentioning the LP upper bound.

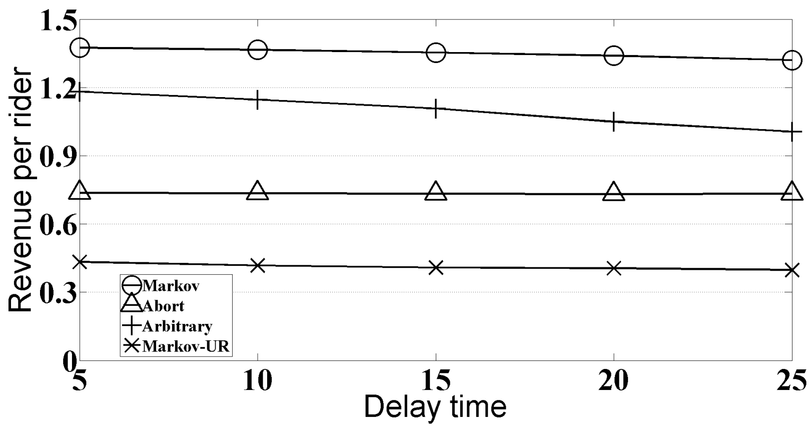
In our experiments, we compared, under execution uncertainty, the performance of our Markov strategy (obtained by solving LP (17)) against two types of benchmarks: game-theoretic, deterministic, policies assuming no execution uncertainty and Markov policies that take execution uncertainty into account, but that assign actions based on a uniform random probability. The pre-generated schedules are calculated using TRUSTS (Yin et al., 2012), the original system developed for the patrolling train lines, based on a deterministic model assuming no execution uncertainty. However, since actions to take after deviations from the original plan are not well-defined in TRUSTS schedules, we augmented these pre-generated schedules with two naive contingency plans indicating the actions to follow after a unit deviates from the original plan. The first plan, "Abort", is to simply abandon the entire schedule and return to the base. The second plan, "Arbitrary", is to pick an action randomly from all available actions at any decision point after the deviation. The uniform random Markov strategy (Markov UR) then assigns, given a state $s \in S_i$ of the MDP defined in Section 3.2, a uniform probability to all the actions taken in s leading to another state $s' \in S_i$. In essence, this strategy is similar to the "Arbitrary" policy but it assumes that resources always pick a random action and not only after they are deviated. It was chosen because this is the approach that security agencies adopt when not using a game-theoretic approach for randomization[6].

Each of these strategies was generated using CPLEX 12.2 with the barrier method on standard 2.8GHz machines with 4GB memory. Then, to generate significant datapoints, each strategy was evaluated using Monte Carlo simulations with 100000 samples. In such simulations, the riders were assumed to choose a best response based on the frequency of being checked over these samples. The discussion of our results follows.

6. See the work of Tambe (2011) for more detail.



(a) Varying uncertainty



(b) Varying delay

Figure 5: Defender's revenue per rider on the Red line: Markov vs. TRUSTS' pre-generated strategies and Markov UR

6.2.1 Expected Revenue against Varying Delay Probability and Time

To run this experiment, we fixed the number of units to 6 and the patrol length to 3 hours and varied the delay probability and the delay time. The results presented below are based on simulations on the Red line. Figure 15 in the Appendix A shows the results for the Blue, Green and Gold lines.

First, we fixed the delay time to 10 minutes and varied the delay probability η from 0% to 25%. As we can see in Figure 5(a), "Abort", "Arbitrary" and "Markov-UR" performed poorly in the presence of execution uncertainty. With increasing values of η, the revenue of "Abort" and "Arbitrary" decayed much faster than the Markov strategy. In fact, by increasing the delay probability, the number of interruptions is also increased. In such situations, both the "Abort" and the "Arbitrary" strategies will perform sub-optimal actions (dropping the schedule or selecting a random action) which will yield a poor performance.

For example, when η was increased from 0% to 25%, the revenue of "Abort" and "Arbitrary" decreased 75.4% and 37.0% respectively while that of the Markov strategy only decreased 3.6%. In contrast, the performance of"Markov-UR" remained constantly around the same value of, approximately 0.4. This is expected, since the strategy constantly performs random actions, therefore its performance will be independent from the delay probability.

An important observation here is that the revenue of "Abort" decayed extremely fast with increasing η — even with a 5% probability of delay, the revenue of "Abort" was only 73.5% of that of the Markov strategy. With a conservative estimate of 6% potential fare evaders (?) and $300,000$ daily riders in the LA Metro Rail system, the 26.5% difference implies a daily revenue loss of $6, 500 or $2.4 million annually.

Second, we fixed η to 10% and varied the delay time from 5 to 25 minutes. The results, in Figure 5(b), present similar trends to the ones in Figure 5(a). The three strategies "Abort", "Arbitrary" and "Markov-UR" performed worse than the Markov strategy. With increasing delay time, the revenue of "Arbitrary", decayed in a faster rate than the Markov strategy. Similar to what discussed earlier, after a delay, the strategy would start to generate sub-optimal actions which would result in a low expected revenue. In contrast, both the revenue of "Abort" and of "Markov-UR" remained the approximately the same. The time of the delay does not matter for the "Abort" strategy if a resource abandon the schedule after the first unexpected event. Similarly, the time of the delay does not matter for "Markov-UR" if a resource only performs random actions. When the delay time was increased from 5 to 25 minutes, the revenue of "Abort" and "Markov UR" remained the same, 0.4 and 0.75 respectively, while that of "Arbitrary" and the Markov strategy decreased 14.4% and 3.6% respectively.

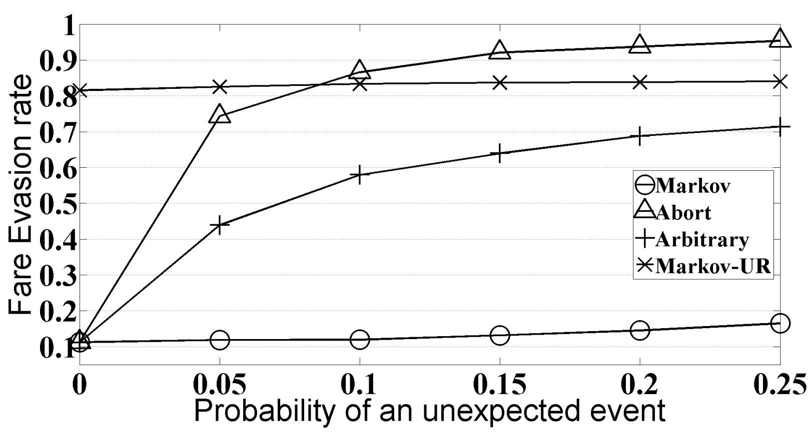
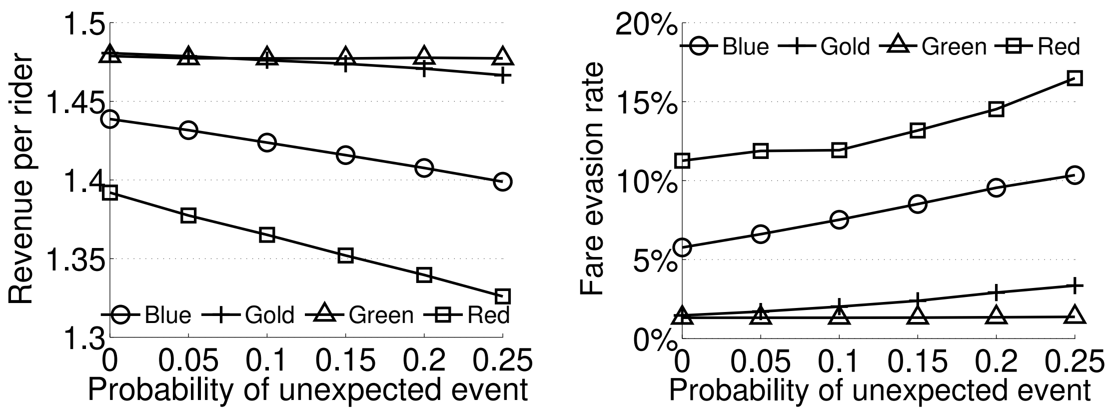


Figure 6: Markov vs. deterministic strategies: evasion rate

6.2.2 Fare Evasion Rate against Varying Delay Probability and Time

The settings for this experiment were the same as the ones for the experiment discussed above. Here, we present the results for the Red line only. The results for the Blue, the Green and the Gold line, present similar trends and are shown in Figure 16 in the Appendix A. As discussed in Section 6.1, we considered a rider to prefer fare evasion if and only if his expected penalty from fare evasion is lower than \$1.5, the ticket price. The results are shown in Figure 6, which shows the fare evasion rate of the four policies with increasing η. As we can see, "Abort", "Arbitrary" and "Markov UR" showed extremely poor performance in evasion deterrence with even a tiny probability of execution error. Similar to the first experiment, increasing the number of interruptions led the two deterministic strategies to produce sub-optimal actions yielding a fewer number of fare evaders captured. In particular, when the delay probability η was increased from 0% to 5%, the evasion rate of the Markov strategy barely increased while that of "Abort" and "Arbitrary" increased from 11.2% both to 74.3% and 43.9% respectively. In contrast, the "Markov UR" strategy remained stable around a fare evasion rate of 80%. This result confirms the trend for the "Markov UR" strategy depicted in Figure 5(a). The delay probability does not affect the performance of the strategy because it consists of computing random actions.



(a) Revenue per rider of Markov strategy

(b) Evasion rate of Markov strategy

Figure 7: Markov strategy over different lines.

6.2.3 Robustness of the Approach Against increasing Delay Probability

To run this experiment, we fixed the number of units to 6 and the patrol length to 3 hours, but varied the delay probability η from 0% to 25%. Figure 7(a) and Figure 7(b) show the expected revenue per rider and the evasion rate of the four lines respectively[7]. As we can see, the revenue decreased and the evasion rate increased with increasing η. However, the Markov strategy was able to effectively allocate resources to counter the effect of increasing η in terms of both revenue maximization and evasion deterrence. For example, the ratio of the revenue of $\eta = 25\%$ to that of $\eta = 0\%$ was 97.2%, 99.1%, 99.9%, 95.3% in the Blue, Gold, Green and Red line respectively. Similarly, when η was increased from 0% to 25%, the evasion rate of the Blue, Gold, Green and Red line was increased by 4.6, 1.9, 0.1, 5.2 percentage points respectively. Thus, the Markov strategy performance degraded gracefully as uncertainty increased for each of the four lines.

7. The revenue of the Red line was significantly lower than the other lines because fare check effectiveness f_λ defined in Section 5.1 was set inversely proportional to the ridership volume.

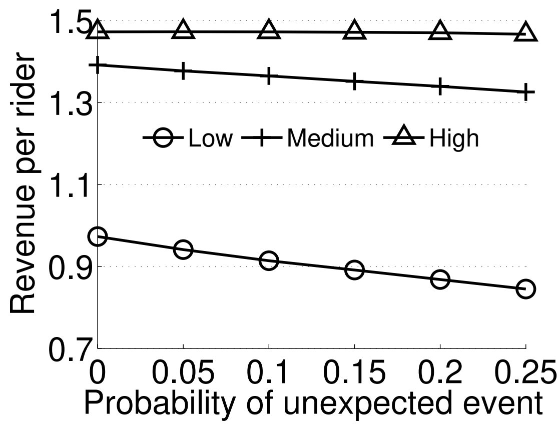


Figure 8: Revenue decay with varying coverage levels

6.2.4 Revenue per Rider while Increasing Delay Probability

In this experiment, we considered 3, 6 and 9 patrol units, representing three levels of fare enforcement: low, medium and high, respectively, and evaluated the revenue per rider with increasing η. The results for the red line are depicted in Figure 8. Results for the blue, the green and the gold line present similar trends and are depicted in Figure 17 in the Appendix A. As shown in Figure 8, the rate of revenue decay with respect to η decreased as we increased the level of fare enforcement from low to high. Intuitively, with more resources, the defender could better afford the time spent on handling unexpected events without sacrificing the overall revenue. For example, when η was increased from 0% to 25%, the revenue drop in the low, medium and high enforcement setting was 13.2%, 4.7%, and 0.4% respectively.

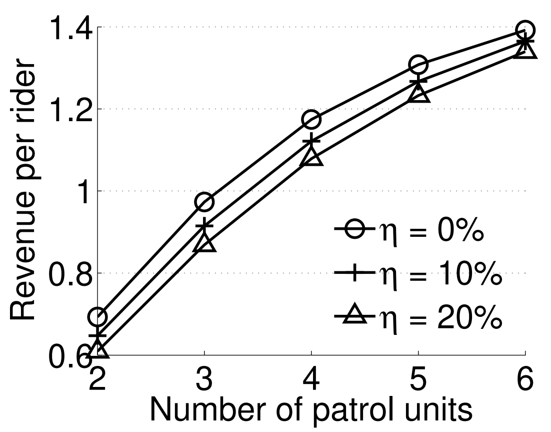


Figure 9: Revenue per rider with increasing coverage

6.2.5 Revenue per Rider while Increasing the Number of Resources

In this experiment, we fixed the patrol length to 3 hours, we then considered three delay probabilities $\eta = 0\%$, 10%, and 20%, each representing an increasing level of uncertainty. To measure the impact of the number of units, we increased the number of units from 2 to 6. The results for the Red line are shown in Figure 9, Figure 18 in the Appendix A shows the results for the Blue, Green and Gold lines. Both figures show the revenue per rider for the defender for increasing number of units. As depicted in the figure, as we increased such number, the revenue increased towards the maximal achievable value of \$1.5 (ticket price). As the number resources increases, the algorithm produces Markov strategies which distribute the resources so as to check more rider types. For example, as shown in the figure, when $\eta = 10\%$, the revenue per rider was \$0.65, \$1.12, and \$1.37 for 2, 4, and to 6 patrol units respectively. In addition, the figure shows that as the uncertainty increases, the revenue per rider slightly decays. For example, considering 4 units, the revenue per rider is 1.09, 1.13 and 1.18 for $\eta = 0\%$, 10% and 20% respectively.

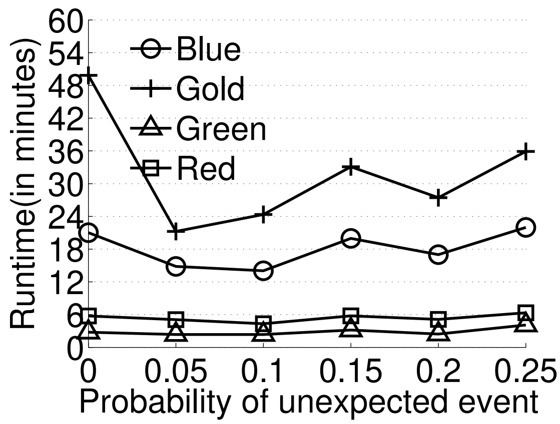


Figure 10: Worst-case LP runtime

6.2.6 Runtime of the LP Algorithm

To confirm this hypothesis, we ran an experiment considering the worst-case runtime (over 10 runs) of the LP with increasing η for the four metro lines. The number of units was fixed to 3 and the patrol length per unit was fixed to 3 hours. To verify whether the delay probability had any impact on the runtime, we ran simulations varying η from 0% to 25%. The results are depicted in Figure 10. As we can see, the algorithm could solve all of the problems within an hour. For example, when $\eta = 10\%$, the runtime for the Blue, Gold, Green, and Red line was 14.0, 24.3, 2.4, and 4.3 minutes respectively.

In addition, these results present a number of additional features that can be analyzed. The runtime varied among the four Metro lines, related to their number of states and types. In other words, the number of stations and daily trains (i.e. the density of the train schedule) affected the runtime of the algorithm. For example, since the Green line has significantly fewer types and states, solving the LP was easier than the other three lines.

A surprising result is the fact that we found no correlation between the runtime and delay probability η. Our results showed that, for all of the four lines, stochastic models with $\eta > 0\%$ took less time to solve than deterministic models ($\eta = 0\%$). More precisely, with increasing η beyond 0%, the runtime of all of the four lines fluctuated and showed an upwards trend, yet the correlation between runtime and delay probability was not statistically significant.

6.3 Real-World Experiment

The results of the simulations presented above showed that deterministic approaches that do not take execution uncertainty into account perform poorly. Given the large number of interruptions, the officers were rarely able to complete a schedule provided by deterministic strategies. As discussed in Section 1, these results motivated this work and led to our new solution concept based on an MDP. In addition, considering the limited time that we were given by the LASD to run our experiment, we decided to use only the "Markov UR" strategy as a benchmark in the real world experiment. Real-world failure was not acceptable for the LASD. Therefore, they recommended not testing deterministic schedules any further. In addition, despite performing poorly in simulation, not only is "Markov UR" the strategy used by most security agencies to automatically allocate their resources, but schedules can be updated whenever an interruption occurs.

The real world experiment took place over 21 days during the months of July and August 2013. The organization of the experiment (e.g., train the security officers, design and organize the experiment in collaboration with the LASD) required approximately two weeks. This experiment had two key purposes. The first was to validate the MDP model in the real world. The second was to run a head-to-head comparison between our game-theoretic approach and the Markov-UR approach. This section discusses the setup of the experiment and the results that we obtained.

6.3.1 Experiment Setup

The fare evasion experiment took place on the Blue line of the LA Metro system (see Figure 11 for the map of the metro line). Other lines could not be tested, because the LASD only allowed us to use our strategies on the Blue line during our real-world test. This line consists of 22 different stations and is one of the biggest lines in the LA Metro system. It was selected by the LASD, which helped to organize the experiment (e.g., assign security officers and patrol times).

Each day, a team of two security officers (see Figure 12), was randomly selected by the LASD, to patrol the line for a duration of at most 120 minutes. Patrols were run during both the morning and the afternoon. Some days the tests ended early due to the officers being reassigned. One of the two officers acted as the leader of the team: he was given the smartphone, he had to read the schedule to the other officers, collect the data and eventually update it whenever a delay occurred. An update could be made either during a station-check as described in Section 6.1) or during a train-check. In the latter case, the officers were required to leave the train at the next station to request an update. This was required because, as discussed in Section 5.1, the Markov strategy was defined over each

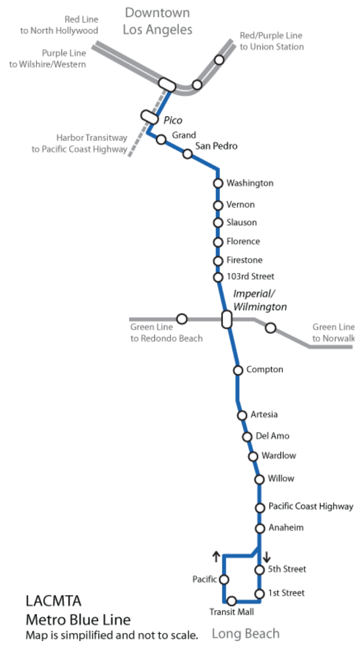


Figure 11: The map of the blue line of the LA Metro

Figure 12: Two security officers performing fare checks on a train.

state of the MDP (i.e., station, time). Thus any new strategy has to be sampled from a specific state. Every week the team was provided with one of two types of schedules:

Game-theoretic schedules (GT): This type of schedule was generated according to the LP in Equation (17) presented in Section 5.1.

Markov UR schedules (UR): This type of schedule was generated by modeling the problem as an MDP, as discussed in Section 3.2. However, the corresponding Markov strategy π_{s_i,a_i}, for each state s_i and action a_i was calculated assuming a uniform probability distribution.

The officers were not told which schedule they were using as not to bias their performance. Before the experiment, we anticipated that the officers might view some of the schedules as leading to low performance in terms of catching very few fare evaders. In such situation, the officers, in order to avoid poor performance, might end up voluntarily deviating from their given schedules to reach a better location because they were unsatisfied with the current one. In anticipation of such voluntary deviations, we augmented both the game-theoretic and UR schedules with the ability to perform updates. More specifically, we allowed the officers to request VOLUNTARY or INVOLUNTARY updates. VOLUNTARY updates consisted of the officers updating the current schedule because in their opinion, the current specified action was not fruitful as a venue to check fares. Officers were allowed to choose a new location that they considered more fruitful for catching fare evaders and request a new schedule from there. INVOLUNTARY updates consisted of the officers requesting a new schedule because they were delayed (e.g., from issuing citations or arresting a suspect) and were unable to perform the next action on their schedule. This type of update could be requested *anytime* an officer was delayed. As we will see below the officers used VOLUNTARY updates almost every day with the UR schedules, but never in the GT schedules.

Finally, it is important to notice that given the duration of our experiment, the game-theoretic schedules are essentially testing a maximin strategy. As discussed in Section 5.1, the LP computes a Stackelberg strategy, a strategy based on the assumption that the riders will conduct surveillance and observe the defender's mixed strategy. However, considering only 21 days of patrol whereby the officers could only patrol less than few hours per day, either in the morning or the afternoon, we cannot assume that the riders had sufficient time to conduct accurate surveillance, observe the mixed strategy and best respond to it. Nonetheless, the LP in Section 5.1 solves a zero-sum game for which a Stackelberg equilibrium and the maximin strategy are known to be equivalent (Yin et al., 2012). Thus, since the maximin strategy provides a guaranteed level of defender utility without making any assumption on the adversary's surveillance of the defender's mixed strategy, these experiments compare the benefit of using a maximin strategy against other (non-game-theoretic) approaches for generating patrol schedules.

6.3.2 Estimating the Uncertainty Parameter

Given the unpredictability of the real-world, two key parameters for instantiating the MDP—the length of a delay and, most importantly, the probability that such a delay could happen— could not be defined before-hand, as was done in Section 6.1 with the remaining problem parameters. In such a setting, adopting a continuous-time MDP could be a possible alternative. However, continuous time MDP algorithms, which involve techniques such as forward search (Marecki & Tambe, 2008; Mausam, Benazera, Brafman, Meuleau, & Hansen, 2005), would appear difficult to implement and would add significant computational complexity. Another alternative, the one adopted in this work, is to adopt a cross validation approach, a well-known technique used in machine learning (Jaulmes et al., 2007) and statistics (Kohavi, 1995). The idea is to select a policy robust against uncertainty, i.e., a policy which guarantees the highest expected revenue in the worst case setting, when uncertainty will minimize the defender's expected utility. To achieve this, we discretized the delays and defined an MDP model which assumed multiple delays, each with a specific probability. We divided this approach into two steps. First, we randomly generated N

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	total
GT	60	60	90	60	90	10	90	110	90	90	105	855
UR	60	60	60	60	60	75	100	100	100	90		765

Table 2: Patrol duration over each of the 21 days.

MDPs. Each MDP is generated assuming that a delay will happen within a time window of 30 minutes. In other words, we assume that resource i can experience delays up to 30 minutes (any delay longer than 30 minutes is considered to be beyond repair and a new schedule is generated). This time window is then divided into five different time intervals: $[0, 6]$, $[6, 12]$, $[12, 18]$, $[18, 24]$ and $[24, 30]$ minutes and one delay is sampled for each interval. In essence, this process discretizes the unknown delay length into 5 possible delays distributed within a 30 minutes time window.

Second, we solve each MDP-based patrolling game using the LP in Section 5.1. In so doing, we obtain N Markov policies π_i^k corresponding to each MDP^k. Next, we cross-validate each policy π_i^k against each $MDP_{k'}$ with $k \neq k'$, i.e., we calculate the expected revenue that each policy π_i^k generates when evaluated against $MDP^{k'}$. This expected revenue is calculated by running 100000 Monte Carlo simulations. Each simulation consists of sampling one policy for the defender and calculate the resulting expected utility against all N MDPs. At the end of the simulations, we obtain a $N \times N$ matrix where the rows represented the policies π_i^k and the columns represent the N MDPs. Each cell (k, k') of the matrix contains the expected revenue obtained by evaluating policy π_i^k against $MDP^{k'}$. Then, the policy to deploy was selected using a maximin strategy. In more detail, we chose the policy which was maximizing the expected utility in the worst case scenario, i.e. considering the MDP yielding the lowest expected utility among all the different MDPs.

From a practical perspective, the policy obtained earlier might not yield a schedule which will represent *exactly* all the delays that might happen during a patrol. However, by modelling five different delays, the schedules are now able to cover a larger range of delays. Hence, whenever an officer is interrupted and requests an update, the smartphone application will simply search the schedule for the state that best matches the officer current location and time and will present the new list of actions starting from there.

6.3.3 Results

During the 21 weekdays of our experiments, we were able to run GT schedules for 11 days of testing while UR schedules were deployed for 10 days, resulting in 855 and 765 patrol minutes, respectively. The schedules were compared using two different metrics. First, we counted the number of passengers checked and the number of captures at the end of each patrol. The captures were defined as the sum of the number of warnings, citations, and arrests. Passengers without a valid ticket could be given a warning or cited for a violation on the discretion of the officer. This metric was chosen because it would allow us to measure the performance of each schedule in the real world. Second, we counted the number of times that the update function was used voluntarily and involuntarily. While involuntary updates helped determine the value of using MDPs as discussed below, voluntary updates measured

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	total
GT	0	1	3	1	1	0	2	2	4	2	1	18
UR	0	2	1	1	1	2	2	2	3	2		16

Table 3: Number of INVOLUNTARY (delays) deviations for each day of patrol

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	total
GT	0	0	0	0	0	0	0	0	0	0	0	0
UR	1	0	1	1	1	0	1	1	1	1		8

Table 4: Number of VOLUNTARY (updates) deviations for each day of patrol

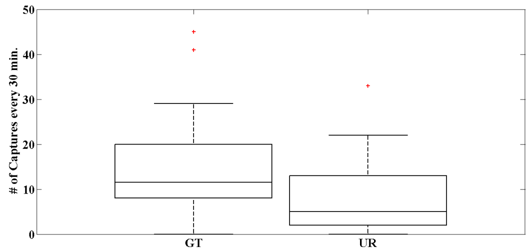
the human (officer) perception of quality of the schedules – the more such voluntary updates, the more the officers were dissatisfied with their given action. Table 2 shows the duration of each day of patrol for both GT and UR schedules[8].

As shown in the table, the actual duration of a daily patrol was often different over the 21 days of the experiment, for both GT and UR schedules. For this reason, providing a comparison normalized over the days of the experiment was impossible. However, most of the days, we were able to collect data for multiples of 30 minutes (e.g., 60, 90 minutes). Hence, to properly compare our results, we divided our data in 30 minutes segments. More specifically, we considered all the train and station checks within a time window of 30 minutes and collected the data resulting from these actions[9]. Having defined the data points, we can now proceed to analyze our results.

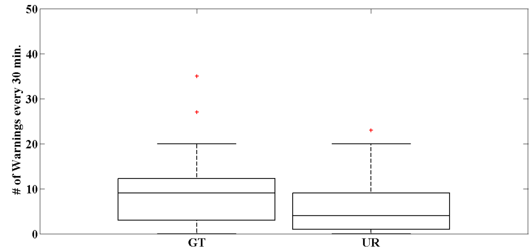
Validation of the MDP model: As discussed at the beginning of this section Both GT and UR schedules were calculated by solving an MDP. For this reason both schedules could be updated to request a new schedule. Tables 3 and 4 then show, for each day of patrol, the number of VOLUNTARY and INVOLUNTARY deviations requested by the officers. In total, GT schedules were updated 18 times, all of which were *involuntary* deviations, i.e., delays. All these update requests confirm that the MDP model was able to provide schedules that could be updated whenever necessary.

All INVOLUNTARY deviations were due to the officers writing citations or helping people. The average delay length was of 12 minutes (the largest delay was of 20 minutes). In each case, as discussed at the beginning of this section, a new schedule was provided starting at the officers' current location and closest time. Finally, Table 4 shows that voluntary deviations were used only with UR schedules. This result strongly suggests that the officers were mostly satisfied with GT schedules. In addition, it means that GT schedules

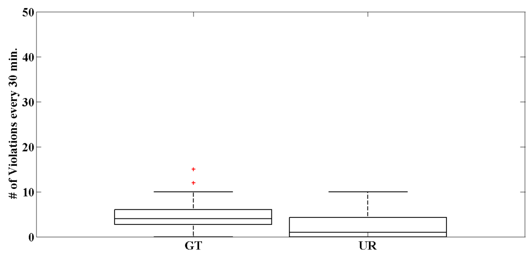
8. As shown in Table 2, each day of patrol correspond to a 2-day test where GT schedules were tested on the first day and UR schedules were tested on the second, both at identical times.
9. In so doing, the segments are also statistically independent. Within each segment the officers will check different people who are unable to affect each other. Each segment corresponds to a sample of different train riders taken at different times and locations. Not only do the officers never check the same rider twice but most importantly, during 30 minutes, they will visit different locations by riding the trains (roughly, one train every 6 minutes in the blue line) and inspecting the stations (on-station operations last no longer than 20 minutes).



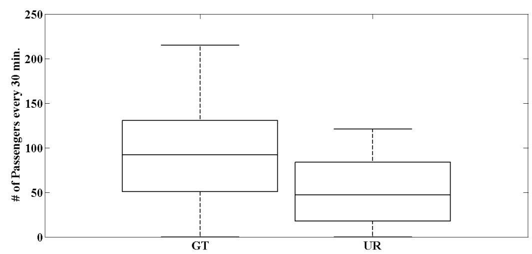
(a) Captures



(b) Warnings



(c) Violations



(d) Passengers

Figure 13: Results of the Fare Evasion tests

did not really compete against UR schedules only. Rather, the comparison was between UR schedules which were augmented with real-time human intelligence for most of the time (8 out of 10 days). We discuss the results of such comparison next.

Game-Theory vs. Uniform Random: The results that we obtained are shown in Figure 13 and in Table 5. Figure 13 shows eight boxplots depicting the data that we collected during each patrol, using both GT and UR schedules. Respectively, the four figures present data collected on captures (Figure 13(a)), warnings (Figure 13(b)), violations (Figure 13(c)), and passengers checked (Figure 13(d)) per 30 minutes of patrolling[10]. For each boxplot, the top and bottom of the box represent the 75th and 25th percentiles, respectively, while the middle line indicates the median of the collected data. The "+" data points indicate statistical outliers, while the whiskers show the most extreme non-outlier data points. Each of the four figures (captures, warnings, violations and passengers checked) shows that the data collected using GT schedules had higher values than the data collected using UR schedules. As shown in Table 5, on average, GT schedules led to, respectively 15.52 captures, 10.42 warnings and 5.03 violations issued every 30 minutes against, respectively against 9.55 captures, 6.48 warnings and 3.07 violations obtained using UR schedules. To confirm the statistical significance of these results, we ran a number of weighted unpaired student t-tests ($p = 0.05$) (Goldberg, Kercheval, & Lee, 2005; Bland & Kerry, 1998) and verified, for each metric, that the difference in the results was statistically significant. We used a weighted t-test because some data segments had a duration shorter than 30 minutes and we wanted to use all the available data for our analysis. As shown in Table 2, not all the patrol durations could be properly divided into a finite number of 30 minutes segments (e.g., UR: D6, D7, D8, D9 and GT: D6, D8, D11). Therefore, we

10. GT schedules also led to two arrests on day 6. This is why the patrol only lasted 10 minutes.

	Days	avg. C	avg. W	avg. V	avg. P
GT	11	15.52	10.42	5.03	96.77
UR	10	9.55	6.48	3.07	60.85

Table 5: Average captures (C), warnings (W), violations (V) and passengers (P) based on the results obtained in Figure 13

calculated a weighted average for each of the metric defined above, whereby each segment was given a weight which was defined based on the segment's duration (longer segments corresponded to higher weights).

From a practical perspective, the magnitude of the difference between the two approaches is significant: cumulatively over a period of 21 days GT would capture a much larger total number of fare evaders. This result can be emphasized even further if we correlate it with the results shown in Tables 4 and 3. While running UR schedules the officers were requesting INVOLUNTARY deviations essentially every day, whereas no such deviations were requested while running GT schedules. In other words, they were using real-time situation awareness to augment the quality of the schedules, thus making the UR schedule more compelling.

The results in Table 5 also indicate that GT schedules led to 96.77 passengers checked every 30 minutes against 60.85 passengers checked by using UR schedules. As discussed in Section 5.1, GT schedules are generated by leveraging all the possible sequences of train and station checks and by taking into account all the key dimensions discussed in Section 6.1, including the train schedules, the officers' effectiveness and, most importantly the daily ridership statistics. This means that stations or trains with a higher presence of riders will be given a higher coverage probability since they are more likely to contain fare evaders. Hence, these results confirm the accuracy of the model as both Figure 13(d) and Table 5 show that GT schedules led the officers to check more passengers than UR schedules.

This raises the question of whether a static type of schedule, which only deploys the officers at the most crowded locations, would lead to similar or even better results than those obtained with GT. Given the limited amount of time that we had to conduct our experiments, we were unable to compare GT schedules against a static deployment – where the key weakness is predictability in the longer term. Indeed, effective randomization was one of the main reasons for LASD to collaborate on these experiments – security agencies know that static schedules become predictable in the long term[11]. After a certain amount of time, the passengers would know where the officers are located and could exploit this information to avoid paying the fare.

7. Summary and Future Work

This paper addressed dynamic execution uncertainty in security resources allocation. More specifically, this paper presented four main contributions. First, we proposed a general Bayesian Stackelberg game model for security patrolling whereby execution uncertainty is

11. Tambe (2011) discusses the benefits of randomization in detail.

handled via a Markov decision process (MDP). Second, we studied the problem of computing a Stackelberg equilibrium (SSE) for this game and showed that by exploiting some structure in the game's utility functions, we could represent the defender's strategy space in a compact form which could be efficiently solved using canonical algorithms for solving Bayesian SSGs. In addition, we showed that it is always possible to generate a mixed strategy with a polynomially-sized support. Third, we ran a number of simulations whereby we tested our framework within various different settings. The key result is that by modeling execution uncertainty as an MDP, we were able to generate policies that overcame the failures of existing SSG algorithms which do not take such uncertainty into account. Finally, for our fourth contribution, we ran a real-world experiment whereby we compared schedules generated using our approach against competing schedules comprised of a random scheduler augmented with officers providing real-time knowledge of the current situation. Our results supported our MDP-based model because we were actually able to show the use of the contingency plans provided by the MDP in the real-world. In addition, our results showed that game-theoretic schedules outperformed the competing schedules, despite the fact that the latter were improved with real-time knowledge. In so doing, these results constitute one of the first examples of the potential of employing algorithmic game theory to solve real-world security allocation problems.

In terms of future work, there exist a number of challenges left to address. One interesting technical challenge is that of addressing adjustable autonomy (Huber, 1999; Scerri, Pynadath, & Tambe, 2002) or mixed initiative planning (Ferguson, Allen, & Miller, 1996) in the context of game theoretic scheduling. Our experiments showed that the officers might deviate from a schedule if they perceive that it might lead to a poor performance in terms of fare evaders captured. Hence, it would be interesting to augment our schedules to take this possibility into account. More specifically, we could extend the game theoretic model described in Section 3 to account for the possibility that the officers would eventually deviate from the schedules and execute some action based on real-time situational awareness.

One interesting empirical challenge would be to run a long-term controlled experiment (e.g., one year) complementary to the one we present in this paper. The idea would be to measure how the riders will react to game-theoretic scheduling. As discussed in Section 6.3, given the practical difficulties related to running real-world security experiments, security agencies such as the LASD typically prefer to avoid running long term experiments because they would interfere with the every-day security of the transit system. Nonetheless, if this could be done, such an experiment would provide some very valuable insight on the effects of SSGs in the real-world.

Acknowledgements

This article is the product of the joint work of Francesco M. Delle Fave and Albert X. Jiang. Both of them are first authors of this work.

In terms of contributions, this article extends the paper by Jiang, Yin, Zhang, Tambe, and Kraus (2013). In this work, we extend this initial version with the following contributions: (i) we present a new theoretical result, whereby we show how we can always calculate an optimal mixed strategy for the defender with a polynomially-sized support; (ii) we extend the simulations presented by Jiang et al. (2013) by evaluating our approach against

a uniform random scheduler; (iii) we present results of a large scale real-world experiment whereby we validate the MDP-based approach defined by Jiang et al. (2013) in the field; (iv) in the same experiment, we compare the actual outcome of executing schedules generated by the MDP-based approach against ones generated using a competing uniform random scheduler — presenting some of the first results of algorithmic game theory in the field; (v) to run these real-world experiments, we describe the development of a smartphone application to load and visualize game-theoretic schedules and a sampling technique to instantiate key parameters of the MDP; (vi) we discuss significant new related work and future work.

We thank the Los Angeles Sheriff's Department for their exceptional collaboration. This research is supported by TSA grant HSHQDC-10-A-BOA19, MURI grant W911NF-11-1-0332 and MOST 3-6797.

Appendix A

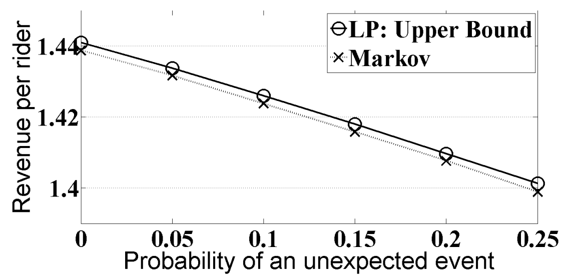
This appendix complements the simulations results discussed in Section 6.2, by presenting the results obtained on the Blue, the Green and the Gold line. Figure 14 compares, for the former three lines, the defender's revenue per rider obtained by the LP (Equation (8)), i.e., the upper bound, and the true value obtained by running 100000 Monte Carlo simulations over the Markov strategy. The experiment was run assuming the same setting as discussed at the beginning of Section 6.2: 3 hours of patrolling and 6 resources.

Figure 15 shows the simulation results complementing the ones presented in Section 6.2 for Hypothesis 1. The setting is the same as described in Section 6.2: 6 resources patrolling the Blue, the Green and the Gold lines for 3 hours with η varying from 0% to 25% and the delay time from 5 to 25 minutes.

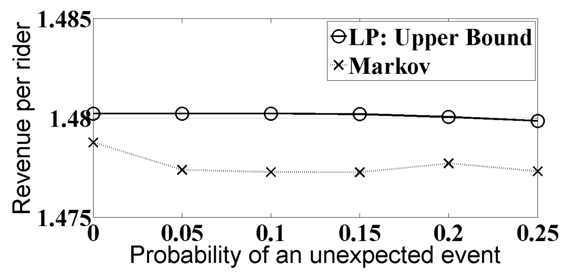
Figure 16 shows the results complementing the ones presented in Section 6.2 for validating hypothesis 2. In this simulation, we considered 6 resources patrolling for 3 hours over each of the four lines and varied uncertainty from 0% to 25%.

Figure 17 shows the results complementing the ones presented in Section 6.2 for validating hypothesis 4. In this simulation, we considered 3, 6 and 9 resources representing 3 levels of coverage, low, medium and high respectively. We then evaluated the revenue per rider varying the delay probability from 0% to 25%.

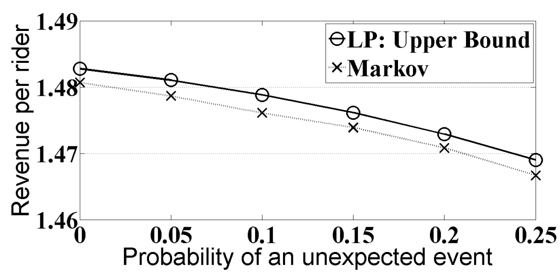
Figure 18 shows the results complementing the ones presented in Section 6.2 for validating hypothesis 5. In this simulation, we considered different delay probabilities (0%, 10% and 20%) and evaluated the revenue per rider varying the number of patrol units from 2 to 6.



(a) Blue line

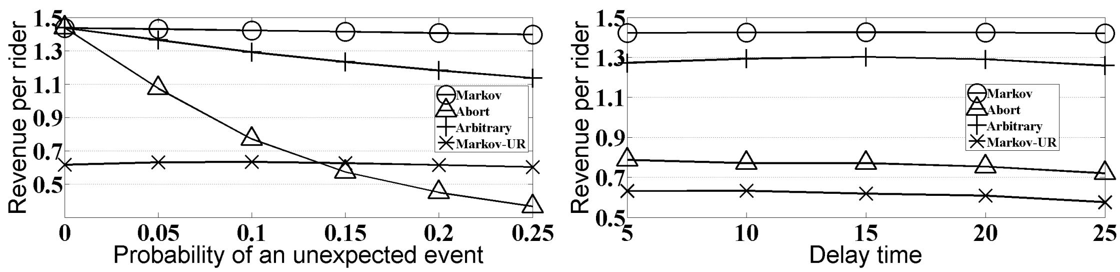


(b) Green line

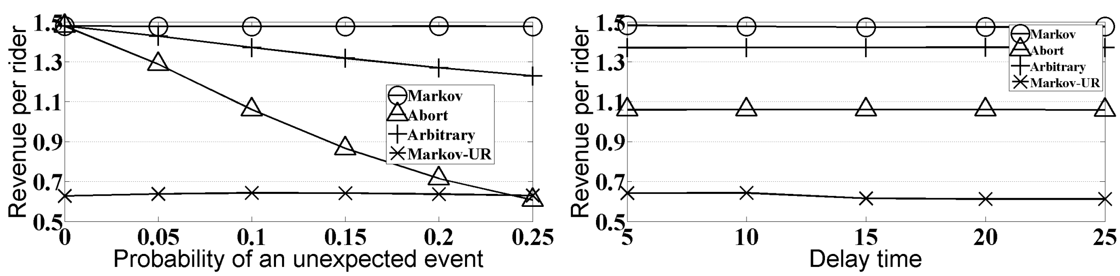


(c) Gold line

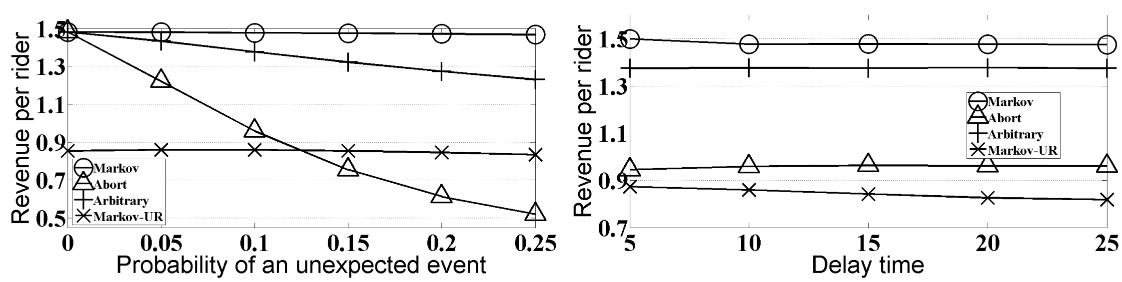
Figure 14: the Markov Strategy (Equation (17)) vs. the true LP (Equation (8)) for the Blue, Green and Gold lines



(a) Blue line

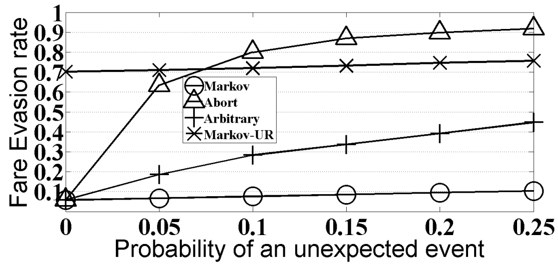


(b) Green line

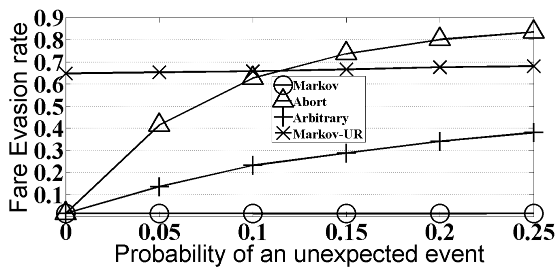


(c) Gold line

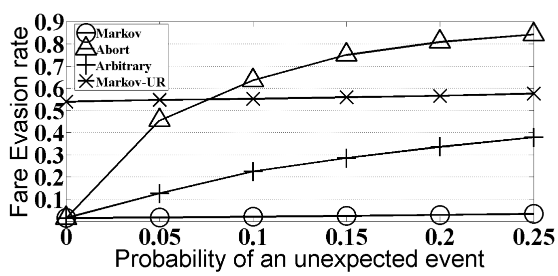
Figure 15: Defender's revenue per rider for the Blue, Green and Gold line, for varying uncertainty and delay time.



(a) Blue line

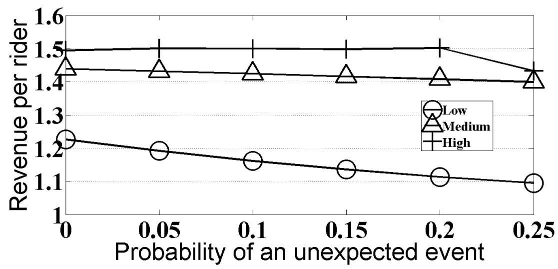


(b) Green line

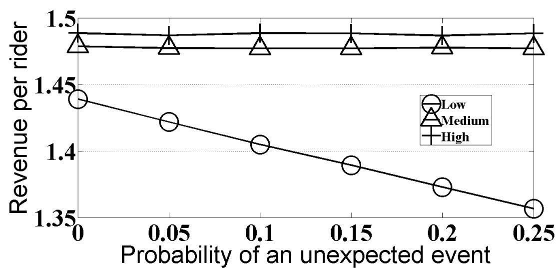


(c) Gold line

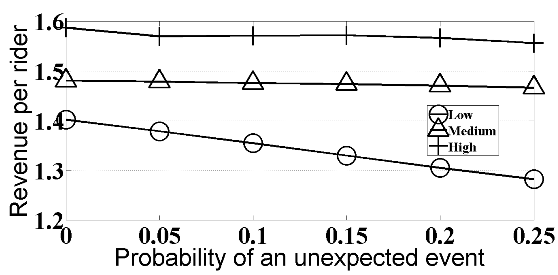
Figure 16: Fare evasion rate over the Blue, Green and Gold lines for varying delay probability



(a) Blue line



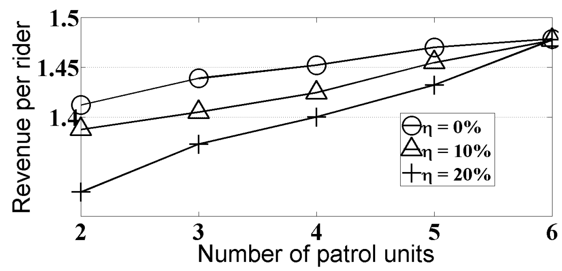
(b) Green line



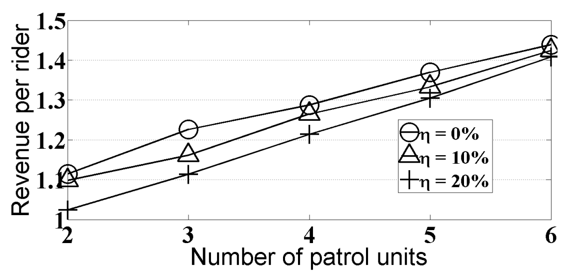
(c) Gold line

Figure 17: Revenue per rider over the Blue, Green and Gold lines for different allocation of resources

(a) Blue line



(b) Green line



(c) Gold line

Figure 18: Revenue per rider over the Blue, Green and Gold lines for three levels of delay probability

References

Agmon, N., Kaminka, G. A., & Kraus, S. (2011). Multi-robot adversarial patrolling: facing a full-knowledge opponent. *Journal of Artificial Intelligence Research (JAIR)*, *42*(1), 887–916.

Agmon, N., Kraus, S., & Kaminka, G. A. (2008a). Multi-robot perimeter patrol in adversarial settings. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 2339–2345. IEEE.

Agmon, N., Sadov, V., Kaminka, G., & Kraus, S. (2008b). The impact of adversarial knowledge on adversarial planning in perimeter patrol. In *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 55–62.

An, B., Kempe, D., Kiekintveld, C., Shieh, E., Singh, S., & Tambe, M. (2012). Security games with limited surveillance. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1241–1247.

An, B., Tambe, M., Ordonez, F., Shieh, E., & Kiekintveld, C. (2011). Refinement of strong Stackelberg equilibria in security games. In *Proceedings of the Twenty-Fifth Conference for the Advancement of Artificial Intelligence (AAAI)*, pp. 587–593.

Aoyagi, M. (1996). Reputation and dynamic Stackelberg leadership in infinitely repeated games. *Journal of Economic Theory*, *71*(2), 378 – 393.

Archibald, C., & Shoham, Y. (2011). Hustling in repeated zero-sum games with imperfect execution. In *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 31–36.

Basilico, N., Gatti, N., & Amigoni, F. (2009a). Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Proceedings of the Eight International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 57–64.

Basilico, N., Gatti, N., Rossi, T., Ceppi, S., & Amigoni, F. (2009b). Extending algorithms for mobile robot patrolling in the presence of adversaries to more realistic settings. In *Proceeding of the Conference of Intelligence Agent Technology (IAT)*, pp. 557–564.

Basilico, N., Gatti, N., & Villa, F. (2010). Asynchronous multi-robot patrolling against intrusions in arbitrary topologies. In *Proceeding of the Conference for the Advancement of Artificial Intelligence (AAAI)*, pp. 345–350.

Becker, R., Zilberstein, S., Lesser, V., & Goldman, C. V. (2004). Solving Transition Independent Decentralized Markov Decision Processes. *JAIR*.

Bland, M. J., & Kerry, S. M. (1998). Weighted comparison of means. *BMJ: British Medical Journal*, *316*, 125–129.

Booz Allen Report (2007). Faregating analysis. Tech. rep., Booz Allen Hamilton Company. Report commissioned by the LA Metro, `http://boardarchives.metro.net/Items/2007/11_November/20071115EMACItem27.pdf`.

Bowling, M., & Veloso, M. (2004). Existence of multiagent equilibria with limited agents. *Journal of Artificial Intelligence Research (JAIR)*, *22*, 353–384.

Brown, A., Camerer, C. F., & Lovallo, D. (2012). To review or not to review? limited strategic thinking at the movie box office. *American Economic Journal: Microeconomics*, *4* (2), 1–26.

Clarke, R. V. (1993). Fare evadion and automatic ticket collection on the london underground. *Crime Prevention Studies*, *1*, 135–146.

Clarke, R. V., Contre, S., & Petrossian, G. (2010). Deterrence and Fare Evasion: Results of a Natural Experiment. *Security Journal*.

Conitzer, V. (2012). Computing game-theoretic solutions and applications to security. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 2106–2112.

Conitzer, V., & Sandholm, T. (2006). Computing the optimal strategy to commit to. In *EC: Proceedings of the ACM Conference on Electronic Commerce*.

Dickerson, J. P., Simari, G. I., Subrahmanian, V. S., & Kraus, S. (2010). A graph-theoretic approach to protect static and moving targets from adversaries. In *Proceedings of the Ninth International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 299–306.

Fang, F., Jiang, A., & Tambe, M. (2013). Protecting moving targets with multiple mobile resources. *Journal of Artificial Intelligence Research (JAIR)*, *48*, 583–634.

Ferguson, G., Allen, J., & Miller, B. (1996). Trains-95: Towards a mixed-initiative planning assistant. In *Proceedings of the 3rd Conference on Artificial Intelligence Planning Systems (AIPS)*, pp. 70–77.

Filar, J., & Vrieze, K. (1996). *Competitive Markov Decision Processes*. Springer.

Gatti, N. (2008). Game theoretical insights in strategic patrolling: Model and algorithm in normal form. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, pp. 403–407.

Goldberg, L. R., Kercheval, A. N., & Lee, K. (2005). t-statistics for weighted means in credit risk modeling. *Journal of Risk Finance*, *6* (4), 349–365.

Grotschel, M., Lovasz, L., & Schrijver, A. (1981). The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, *1* (2), 169–197.

Gruber, P. M. (2007). *Convex and Discrete Geometry*. Springer.

Huber, M. J. (1999). Considerations for flexible autonomy within bdi intelligent agent architectures. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 431–438.

Jain, M., Kardes, E., Kiekintveld, C., Tambe, M., & Ordonez, F. (2010). Security games with arbitrary schedules: A branch and price approach. In *AAAI*.

Jaulmes, R., Pineau, J., & Precup, D. (2007). A formal framework for robot learning and control under model uncertainty. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 2104–2110. IEEE.

Jiang, A. X., Yin, Z., Zhang, C., Tambe, M., & Kraus, S. (2013). Game-theoretic randomization for security patrolling with dynamic execution uncertainty. In *Proceedings of the Twelfth International Conference on Autonomous Agents and Multiagent Systems*, pp. 207–214.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1137–1143.

Korzhyk, D., Conitzer, V., & Parr, R. (2011a). Security games with multiple attacker resources. In *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 273–279.

Korzhyk, D., Conitzer, V., & Parr, R. (2011b). Solving stackelberg games with uncertain observability. In *Proceedings of the Tenth International Conference on Agents and Multi-agent Systems (AAMAS)*, pp. 1013–1020.

Letchford, J., & Conitzer, V. (2013). Solving security games on graphs via marginal probabilities. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 591–597.

Letchford, J., & Vorobeychik (2013). Optimal interdiction of attack plans. In *Proceedings of the Twelfth International Conference of Autonomous Agents and Multi-agent Systems (AAMAS).*, pp. 199–206.

Letchford, J., & Conitzer, V. (2010). Computing optimal strategies to commit to in extensive-form games. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*, pp. 83–92.

Letchford, J., MacDermed, L., Conitzer, V., Parr, R., & Isbell, C. L. (2012). Computing optimal strategies to commit to in stochastic games. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Marecki, J., & Tambe, M. (2008). Towards faster planning with continuous resources in stochastic domains. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, No. 1049–1055.

Mausam, Benazera, E., Brafman, R. I., Meuleau, N., & Hansen, E. A. (2005). Planning with continuous resources in stochastic domains. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pp. 1244–1250.

Nguyen, T. H., Yang, R., Azaria, A., Kraus, S., & Tambe, M. (2013). Analyzing the effectiveness of adversary modeling in security games. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 718–724.

Ostling, R., Wang, J., Tao-yi, J., Chou, E. Y., & Camerer, C. F. (2011). Testing game theory in the field: Swedish lupi lottery games. *American Economic Journal: Microeconomics*, *3*(3), 1–33.

Paruchuri, P., Pearce, J. P., Marecki, J., Tambe, M., Ordonez, F., & Kraus, S. (2008a). Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games. In *AAMAS*.

Paruchuri, P., Pearce, J. P., Marecki, J., Tambe, M., Ordonez, F., & Kraus., S. (2008b). Playing games for security: An efficient exact algorithm for solving bayesian stackelberg games. In *Proceedings of the Seventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 539–547.

Pita, J., Jain, M., Western, C., Portway, C., Tambe, M., Ordonez, F., Kraus, S., & Paruchuri, P. (2008). Deployed ARMOR protection: The application of a game theroetic model for security at the los angeles international airport. In *Proceedings of the Seventh Internation Conference on Autonomous Agents and Multi-Agent Systems (AAMAS).*

Pita, J., Tambe, M., Kiekintveld, C., Cullen, S., & Steigerwald, E. (2011). GUARDS - game theoretic security allocation on a national scale. In *Proceedings of the Tenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 37–44.

Scerri, P., Pynadath, D. V., & Tambe, M. (2002). Towards adjustable autonomy for the real-world. *Journal of Artificial Intelligence Research (JAIR)*, *17*, 171–228.

Shieh, E., An, B., Yang, R., Tambe, M., Baldwin, C., DiRenzo, J., Maule, B., & Meyer, G. (2012). Protect: A deployed game theoretic system to protect the ports of the united states. In *AAMAS*.

Sless, E., Agmon, N., & Kraus, S. (2014). The impact of adversarial knowledge on adversarial planning in perimeter patrol. In *Proceedings of the Thirteenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, p. In press.

Tambe, M. (2011). *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned.* Cambridge University Press.

Tsai, J., Rathi, S., Kiekintveld, C., Ordóñez, F., & Tambe, M. (2009). IRIS - a tool for strategic security allocation in transportation networks. In *Proceedings of the Eight International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 831–839.

Vanek, O., Jakob, M., Lisy, V., Bosansky, B., & Pechoucek, M. (2011). Iterative game-theoretic route selection for hostile area transit and patrolling. In *Proceedings of the Tenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pp. 1273–1274.

Varakantham, P., Lau, H. C., & Yuan, Z. (2013). Scalable randomized patrolling for securing rapid transit networks. In *Proceedings of the Conference for Innovative Applications for Artificial Intelligence (IAAI)*, pp. 1563–1568.

Vorobeychik, Y., & Singh, S. (2012). Computing stackelberg equilibria in discounted stochastic games. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1478–1484.

Weidner, R. R. (1996). Target-hardening at a new york city subway station: Decreased fare evasion– at what price?. *Crime Prevention Studies*, *6*, 117–132.

Yang, R., Kiekintveld, C., Ordonez, F., Tambe, M., & John, R. (2011). Improving resource allocation strategy against human adversaries in security games. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 458–464.

Yin, Z., Jiang, A., Johnson, M., Tambe, M., Kiekintveld, C., Leyton-Brown, K., Sandholm, T., & Sullivan, J. (2012). Trusts: Scheduling randomized patrols for fare inspection in transit systems. In *Proceedings of the Conference on Innovative Applications for Artificial Intelligence (IAAI)*, pp. 59–72.

Yin, Z., Jain, M., Tambe, M., & Ordonez, F. (2011). Risk-averse strategies for security games with execution and observational uncertainty. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pp. 758–763.

Yin, Z., & Tambe, M. (2012). A unified method for handling discrete and continuous uncertainty in Bayesian stackelberg games. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 234–242.