SANDIA REPORT

SAND2012-7179
Unlimited Release
Printed September 2012

Strategic Analysis of Complex Security Scenarios

Yevgeniy Vorobeychik

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone:	(865) 576-8401
Facsimile:	(865) 576-5728
E-Mail:	reports@adonis.osti.gov
Online ordering:	http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone:	(800) 553-6847
Facsimile:	(703) 605-6900
E-Mail:	orders@ntis.fedworld.gov
Online ordering:	http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2012-7179
Unlimited Release
Printed September 2012

Strategic Analysis of Complex Security Scenarios

Yevgeniy Vorobeychik
Scalable Modeling & Analysis
Sandia National Laboratories, P.O. Box 969, Livermore, CA 94551-0969

Abstract

Many advanced technical tools are available to prevent attacks on national infrastructure. Nevertheless, while traditional analyses of security problems have succeeded in producing good technical solutions, they have often ignored the human factor integral to these problems. Human attackers (who may be individuals or state-level attackers) expend substantial effort to breach security because they have the incentive for doing so. People involved in implementing security follow individual incentives, which need not align with global security concerns; consequently, desired security solutions are often implemented poorly, or not at all. This complex interplay between individual incentives and global (organizational and/or national) goals can be modeled and analyzed using game theoretic techniques. By analyzing not only what is possible, but also what is motivated, a holistic approach to security problems can be developed, informing policy and providing tools to policy makers.

We study game theoretic models that unify several current incentive-based approaches to security, and develop simulation-based and mathematical optimization methods for analyzing such models that exploit the high-performance computing capabilities at Sandia. Our first model studies security in interdependent settings, offering a scalable local search heuristic to approximate optimal security decisions in general, and a linear programming approach, coupled with simulations of consequences, to optimally compute security in an important special case. Our second class of models addresses security patrolling problems when an adversary gets to observe the patrol location. We present a general framework, based on stochastic games, for computing optimal security policies in such settings, and present more scalable tools that apply in the important special cases. Our third contribution is a model of security that involves many defenders, but only models non-adaptive attackers (or natural disasters, inadvertent errors, etc). In this model, we demonstrate that the security decisions of many players result in global security configuration that is not very far from optimal, and is much more resilient to environment changes that an optimal solution. This positive effect dissipates, however, when the number of decision makers becomes too large.

Acknowledgments

Most of the work described here was performed in collaboration with a diverse group of people. Chapter 3 was a joint effort with Joshua Letchford from Duke University. Chapter 4 was a collaboration with Milind Tambe and Bo An from USC and Satinder Singh from the University of Michigan. Chapter 5 was a collaboration with Sandians Jackson Mayo, Rob Armstrong, and Joe Ruthruff. A portion of Chapter 3 was published in the Conference on Uncertainty in Artificial Intelligence, 2012, and I would like to thank anonymous referees for their comments and suggestions. A portion of Chapter 4 was published in the National Conference on Artificial Intelligence, 2012, and I would like to thank the referees for their comments, and Vincent Conitzer (from Duke University) from demonstrating that our original claim about stationary Markov policies was incorrect and contributing to its correction. The material in Chapter 5 was published in Physical Review Letters, 2011, and I would like to thank the anonymous referees for their comments and suggestions. The material in Chapter 6 benefited from discussions with Noel Nachtigal, Alyson Kubota, Rob Armstrong, Jackson Mayo, Robert Clay, Susanna Gordon, and Art Hale. Finally, I wish to thank Robert Clay, Keith Vanderveen, and Bob Hutchinson for strongly supporting and encouraging this line of work.

Contents

List of Figures

Chapter 1

Introduction

Securing critical infrastructure has long been at the forefront of research at the National Security Laboratories, including Sandia. More often than not, however, approaches to security fall into two categories. The first class of approaches uses human intuition and expert opinion to identify possible vulnerabilities (e.g., through red teaming exercises) and patch them if feasible and cost-effective. In its more formal incarnations, this approach introduces some structure to such a decision process, involving, for example, the construction of attack graphs, as is done in the IDART methodology. The second class of approaches appeals to statistics to estimate the likelihood of security breaches based on past data, or, as is done in intrusion and spam detection, may use machine learning techniques to attempt to predict security threats. Neither of these approaches takes seriously the fundamental reality that security breaches due to deliberate attacks involve a decision maker (the attacker) who will surely respond to mitigation strategies by consciously changing his attack approach, often actively circumventing whatever security policies are in place. To a certain extent, repeatedly red teaming a system as mitigations and patches are introduced to fix previously discovered vulnerabilities does aim to do precisely this: the red team keeps attacking the system and, ideally, looks to circumvent the applied fixes. However, repeated red teaming efforts are expensive, as they involve highly trained expertise and, moreover, take a considerable amount of time to apply even once. Moreover, red teaming efforts hinge on the human factor: if the same people are used through multiple of patch and red team phases, they may well have reached the limit of their creativity for that specific system early in the process.

In this report, we advocate instead using the formal framework provided by game theory to reason about and anticipate attacker's response to mitigations, and to prescribe mitigation (defense) policies that account for the adversary's response. Specifically, we study game theoretic models that unify several current incentive-based approaches to security, and develop simulation-based and mathematical optimization methods for analyzing such models that exploit the high-performance computing capabilities at Sandia. Our first model (Chapter 3) studies security in interdependent settings, offering a scalable local search heuristic to approximate optimal security decisions in general, and a linear programming approach, coupled with simulations of consequences, to optimally compute security in an important special case. Our second class of models (Chapter 4) addresses security patrolling problems when an adversary gets to observe the patrol location. We present a general framework, based on stochastic games, for computing optimal security policies in such settings, and present more scalable tools that apply in the important special cases. Our third contribution (Chapter 5) is a model of security that involves many defenders, but only models non-adaptive attackers (or natural disasters, inadvertent errors, etc). In this model, we demonstrate that the security decisions of many players result in global security configuration that is not very far from optimal, and is much more resilient to environment changes that an optimal solution. This positive effect dissipates, however, when the number of decision makers becomes too large.

In the final chapter (Chapter 6) we step back to study the concepts of trust and risk from both a decision theoretic and game theoretic perspectives. In this chapter, we draw analogies between our framework for understanding risk and the typical heuristic methods used in much practice, and elucidate the distinction between decision and game theoretic approaches.[1]

[1] The work described in chapters 3, 4, 5, and 6 was jointly supported by the Advanced Simulation and Computing program.

Chapter 2

Game Theory and Security

2.1 Game Theory, In General

Over the years, game theory has received much attention from experts and non-experts alike. A typical layman, and media, perception of game theory is at least fifty years antiquated, for better of for worse, and this is all the more true in the context of security. Among experts, opinions vary depending on the field of expertise. For economists, who are generally fond of constructing models of the world based on the "rational man" (or homoeconomicus) hypothesis, game theory is a natural fit, as its mathematical foundations are firmly embedded in optimization. For many other social scientists, in contrast, game theory has been a common target of scorn, as its mathematical assumptions are demonstrably at odds with data about actual human behavior. In this author's view, many attacks on game theory are founded in poor or antiquated understanding of the field and its goals. My aim in this section, therefore, is both to provide formal definitions of the specific game theoretic concepts that will be used later on, and a discussion about their limitations, real and mythical.

2.1.1 Games

Game theory is, first and foremost, a *formal mathematical model of strategic interactions*. Specifically, it is a way for us to description in a stylized way (that is, leaving only details that are most salient to the goals at hand) who the *players* (interested and empowered parties) are, what they can do (i.e., player *actions* or *strategies*), and what they care about (i.e., *payoff* or *utility* functions). We say that there is a set I of players, and each player is then denoted by $i \in I$; throughout, we will assume that there are finitely many players, and let $n = |I|$ be the number of players. (Indeed, in most cases, there will only be 2 players.) To player i be assign a set of choices A_i. There is varying terminology used for the set of player choices in the game theory literature. Sometimes it is called "actions"; other times "strategies". There is also the possibility, at the core of crucial mathematical concepts within game theory, that players randomize their decisions, meaning that they non-deterministically decide which of their choices to follow; each probability distribution over choices is then itself considered a strategy (since this is something that a player may do). Much confusion may result from these various terms, even while conceptually distinctions may often be entirely unimportant. We will use the following terminology. When we say "actions", we intend to mean the most basic choices available to the player in a particular definition of the game. Commonly, these are called *pure strategies*. When we say “strategies”, we mean any set of choices, deterministic or randomized, available to players in the game; these could, for example, include probability distributions over actions, in which case they are often called *mixed strategies*. The set of strategies of a player i will be denoted by S_i. The crucial distinction between game theory and optimization is that when there are at least 2 players, decisions by all players may matter for each of them. Let $A = A_1 \times \cdots \times A_n$ be the set of joint player actions (frequently called *action profiles*) and, similarly, let $S = A_1 \times \cdots \times S_n$ be the set of joint player strategies (*strategy profiles*). Then, we will use $a \in A$ to refer to a particular vector of actions, one for each player, and $s \in S$ will be a vector of player strategies. An extremely confusing piece of notation that has become universal in game theory is to allow a_{-i} (resp. s_{-i}) refer to the vector of actions (resp. strategies) of all players *not including player i*; thus, for example, $a = (a_i, a_{-i})$, that is, a vector of all players' actions is just a combination of i's action, and actions by everyone other than i.

Finally, preferences of players are typically expressed with respect to joint actions a, or strategies, s. Each player is endowed with a utility function, $u_i(a)$ in the former case, and $u_i(s)$ in the latter. Indeed, in most cases the utility function is actually defined only with respect to player actions, and subsequently *extended* to randomized strategies by taking expected utility with respect to the joint probability distribution which mixed strategies come to represent; that is:

$$u_i(s) = \sum_{a \in A} s(a) u_i(a),$$

where $s(a) = \prod_i s_i(a_i)$, with $s_i(a_i)$ representing the probability that action a_i is played by player i under mixed strategy s_i (thus, we literally equate a player's mixed strategy with a probability measure). This completes the formalism of games in so-called *normal form*.

A few comments are in order at this point. It is easy to miss how general the above construction (i.e., the normal form) is. For example, in taking the sum when we define $u_i(s)$ we imply that A is finite. This is done to avoid unnecessary complexity at this point, but let us assure the reader that $u_i(s)$ can similarly be extended where a are finite-dimensional vectors. Moreover, ignoring the technical distinctions between actions and strategies, we can define games as above where player strategies are *functions*. Thus, for example, a player may condition his decision on time and all past observations that are relevant to his decision. The point is that the above formalism is in no way fundamentally static. Furthermore, if there is uncertainty about outcomes for any particular choice of player actions a, we can fold that into the definition of the corresponding utility functions; hence, the above construction also allows us to model situations where there is uncertainty. Because of its generality, therefore, the formalization of games we presented thus far allows considerable leverage. Nevertheless, we shall see below that there are other, more detailed, formulations which allow us to explicitly represent salient aspects of strategic scenarios (games) in order to better focus our analysis.

There is a special class of games, referred to variously as *zero-sum*, *constant-sum*, and *strictly competitive*. These games have only 2 players, with interests that are completely opposed; thus, when one of these gains, the other necessarily loses. Formally, a zero-sum game is defined exactly as above, with the added caveat that $u_1(a) + u_2(a) = c$ for all $a \in A$, where c is an arbitrary constant (for example, 0, hence the term zero- or constant-sum; the value of the constant is mathematically irrelevant, although it may well be relevant from the perspective of framing and loss aversion in practice).

2.1.2 Equilibria

Upon being handed a game model, we must still determine what to do with it. In optimization and decision theory, it is relatively straightforward: you have an objective function, perhaps some constraints, and you wish to make the best decision. In game theory, the notion of a best (or optimal) decision for a player is ill-defined, since it depends on what others do. This fundamental indeterminacy is, indeed, what makes game theory in general very difficult to apply in practice, even if one believes its assumptions of perfect rationality. This indeterminacy, however, does not emerge in zero-sum games: these "work" very much like optimization problems. In a formal sense, if the action sets are finite and we consider mixed strategies, each player should unambiguously choose a strategy which minimizes the other's maximum utility (i.e., a minimax strategy). In a landmark result, von Neumann [108] proved the following theorem:

Theorem 2.1.1. *For every zero-sum finite game,* $\min_{s_2} \max_{s_1} u_1(s_1, s_2) = \max_{s_1} \min_{s_2} u_1(s_1, s_2)$.

Let us call any pair of player strategies in a zero-sum game, (s_1, s_2), a *solution* if, jointly, they satisfy the condition in Theorem 2.1.1; equivalently, we will also call it an *equilibrium*. Zero-sum games are, thus, supremely elegant: the order of moves makes no difference (as long as players cannot observe actual realizations of each other's actions, but only the mixed strategies), and even if there are many alternative solutions, they are "interchangeable", that is, taking the strategy of one player from one solution and plugging in a strategy of the other player from a different solution together still makes a pair of strategies which satisfy the equality in Theorem 2.1.1. We shall make use of some of these properties later on.

Elegant or not, zero-sum games are patently unrealistic in most interesting settings (save, perhaps, two-player boardgames like chess). As Shelling [100] argued, even conflicts are rarely purely competitive: there may well be outcomes that are best, or worst, for both parties; so, for example, both the USA and the USSR had an incentive to cooperate, even if tacitly, to avoid nuclear war. The natural question, then, is: what is a "natural" solution for games which are not strictly competitive? This question was answered by John Nash, who first defined what came to be called a *Nash equilibrium* as follows:

Definition 2.1.1. A strategy profile s is a Nash equilibrium if

$$u_i(s_i, s_{-i}) = \max_{s_i' \in S_i} u_i(s_i', s_{-i}) \quad \forall i \in I.$$

In words, a Nash equilibrium profile is composed of strategies which are mutual best responses (put differently, each player is doing the best he can given that others stay put). Nash then articulated the following groundbreaking result:

Theorem 2.1.2. *Every finite game has at least one Nash equilibrium in mixed strategies.*

We mentioned earlier that the normal form representation of games is, in a sense, very general. It pays, however, to extend this representation in a number of cases to allow us to analyze games possessing specific properties. One such property of enormous importance is uncertainty; we mean, in particular, uncertainty that players have regarding each other's utility function. To formally define a game that captures this property we make use of a notion of a player's *type*, which is just a mathematical construct allowing us to conveniently index a player's utility function. Let's take an arbitrary player i and suppose that i has a type $\theta_i \in \Theta_i$ (where Θ_i is the set of all possible types of i). Then his utility function is $u_i(a, \theta_i)$ where $a \in A$ is the joint set of player actions, as before. We may now suppose that each player knows that if i has type θ_i, his utility function is $u_i(a, \theta_i)$; the caveat is, players are uncertain about i's type, aside from i, of course, who certainly knows his own type. Next, we introduce a probability distribution $F_i(\theta_i)$ over i's types, which we also posit is common knowledge. For simplicity, assume that each player's type is drawn independently, although we make this assumption only to simplify exposition a bit and it is natural to generalize it.

An important thing to note at this point is that since players other than i do not know i's type, they must reason about his strategy as if i could have *any possible type*; that is, i's strategy is, for their purposes, a function $a_i(\theta_i)$ of i's type (i's mixed strategy is a probability measure over such functions, but let's not worry about that for the moment). The following then, is an extension of the Nash equilibrium solution concept which explicitly accounts for all the constructs in what is called either a *Bayesian game*, or a *game of incomplete information*.

Definition 2.1.2. A strategy profile $(a_1(\theta_1), \ldots, a_n(\theta_n))$ is a *Bayes-Nash equilibrium* if

$$E_{\theta_{-i} \sim F_{-i}}[u_i(a_i(\theta_i), a_{-i}(\theta_{-i}), \theta_i)] = \max_{a_i' \in A_i} E_{\theta_{-i} \sim F_{-i}}[u_i(a_i', a_{-i}(\theta_{-i}), \theta_i)] \quad \forall i \in I, \theta_i \in \Theta_i(a.s.).$$

(The a.s. designation means that we only care about a subset of types here which occurs with probability one, and the condition may not hold for a subset that occurs on a probability zero subset of types).

Now we turn our attention to the final significant game theoretic construct that will serve our purposes later on: Stackelberg games. At the moment, the game definitions involved no dynamics whatsoever. Stackelberg games incorporate dynamics of the minimal sort: they involve two players, one moving after the other. The significance of sequencing moves is that the *follower* (i.e., the chap, or lady, that moves second) *observes* the decision by the *leader* (i.e., the fellow that gets to move first). Here it is paramount to explicate precisely what it is that the follower observes. There are, indeed, two options: first, the follower may observe that actual *action* the leader takes, even if the leader may deliberately attempt to randomly select an action; second, the follower may observe the leader's *strategy*, but if the leader randomizes, the follower only knows the corresponding probabilities, *but does not observe actual realizations*. The latter possibility is, in fact, distinctly powerful: the leader in a game possessing such a quality can do no worse than in any Nash equilibrium of a game in which both players move simultaneously! On the other hand, if the follower observes actual realizations of the leader's actions, it is he that may have an advantage. In most of our work below we, in fact, assume that the follower can only react to the leader's strategy, but not directly to actual realizations, should the leader randomize. For the purposes of analyzing Stackelberg (and, later, security) games, we introduce now the appropriate solution concept for such settings, called a *Stackelberg equilibrium*. First, it is convenient to define a set of best responses for the follower.

Definition 2.1.3. We say that s_f is a best response to s_l if

$$u_f(s_l, s_f) = \max_{s'_f \in S_f} u_f(s_l, s'_f).$$

Let $BR(s_l)$ be the set of all best responses to s_l. Let $BRL(s_l)$ be the set of all leader-optimal best responses, that is, $s_f \in BRL(s_l)$ if $s_f \in BR(s_l)$ and

$$u_l(s_l, s_f) = \max_{s'_f \in BR(s_l)} u_l(s_l, s'_f).$$

Let $BRP(s_l)$ be the set of all leader-pessimal best responses, that is, $s_f \in BRL(s_l)$ if $s_f \in BR(s_l)$ and

$$u_l(s_l, s_f) = \min_{s'_f \in BR(s_l)} u_l(s_l, s'_f).$$

Definition 2.1.4. The profile of leader-follower strategies (s_l, s_f) constitutes a *Stackelberg equilibrium* if s_f is a best response to s_l, or, formally,

$$s_f \in BR(s_l),$$

and s_l is the optimal strategy for the leader, that is

$$s_l = \max_{s'_l \in S_l} u_l(s_l, f(s_l)),$$

where $f(s_l) \in BR(s_l)$ for all s_l. It is a *Strong Stackelberg equilibrium* or *SSE* if $f(s_l) \in BRL(s_l)$ above, and a *Weak Stackelberg equilibrium* or *WSE* if $f(s_l) \in BRP(s_l)$ above.

A SSE in the definition above means, in essence, that the follower breaks ties in leader's favor, whereas a WSE involves a follower who breaks ties to maximally hurt the leader. Surprisingly enough, it is SSE that is usually used in the literature on security games. There are two reasons. The first is that SSE is guaranteed to exist, while a WSE is not. The second is that if the follower only observed a mixed strategy of the leader, the leader can make a tiny change to his SSE strategy, with negligible effect on his own payoff, but ensuring that the follower now strictly prefers his corresponding SSE response s_f to any other. As these nuances are, more often than not, relatively insignificant in practice, we will not concern ourselves with them very much here.

At this point, we have built up the necessary foundations to transition to the problem of formally modeling security using game theory.

2.2 Security Games

Security is a kind of catch-all term, often with a wide variety of meanings depending on context. It is, thus, something of a challenge to create a general framework of any kind to model security.

We will attempt to do so anyway. To us, security will mean an encounter among *defenders*, or parties that wish to prevent negative effects on assets that they possess, and *threats*, or sources of negative outcomes. This language is quite generic: a threat can be a lightning strike, a hurricane, or a malicious hacker, and a defender can be an individual with a computer, a firm, or a nation. Often, the language of security implies specifically *malicious threats*, or *attackers*, and this is the perspective that will permeate much of what is to follow. Formally and semantically, we shall distinguish two kinds of threats: a malicious threat, or an attacker, who aims to actively do damage to the defenders, and *nature*, which is meant to model a non-malicious threat, for example human error or natural disasters. Nature and attackers are different quite fundamentally, as the former can be modeled as a probability distribution which does not actively respond to mitigations, whereas the attacker does, in fact, take into account mitigation (defense) strategies in forming its attack vectors.

Through much of our work below, we will make a strong simplification, though one which is rather common, that there is a single defender and a single attacker (or nature). However, this is not necessarily as dramatic as it may seem at first. For example, there may be many different attackers, but only a single attacker would actually deploy an attack at any given point in time; in this case, it suffices to posit a single attacker agent with many different *types*, and use the formalism of Bayesian games to study such settings.

2.3 Stackelberg Security Games

Much of the material that follows in the upcoming chapters fundamentally builds on the now well established line of research on Stackelberg models of security. The Stackelberg models of security are many and varied, but it will suffice for our purposes to describe one model in particular which possesses considerable structure and allows highly scalable SSE computation. This model is referred to simply as *Stackelberg security game* [66].

A Stackelberg security game consists of two players, the leader (defender) and the follower (attacker), and a set of possible targets. The leader can decide upon a randomized policy of defending the targets, possibly with limited (but costless) defense resources. The follower (attacker) is assumed to observe the randomized policy of the leader, but not the realized defense actions. Upon observing the leader's strategy, the follower chooses a target so as to maximize its expected utility.

Formally, the set of (possible) targets (of attack) is denoted by T, with $|T| = n$. The defender has K homogeneous resources, that is, any defense resource can be used to defend any of the targets. If a target $t \in T$ is covered (defended), and then attacked, the utility to the defender is U_t^c, while the attacker's utility is V_t^c. On the other hand, an uncovered (undefended) target is valued at U_t^u by the defender and V_t^u by the attacker if this target is attacked. This notation implies that the utility functions of the defender and attacker only depend on which target is attacked and whether or not it is defended. We say that in this case the targets are *independent*. Crucially, observe that the game between the attacker and defender here is not zero-sum, since relative utilities may well differ for different targets. Let q_t denote the probability that a target t is defended (covered); these will be the decision variables for the defender. Since the attacker faces a decision problem given a fixed q_t, it suffices to consider only deterministic strategies for him.

Kiekintveld et al. [66] offer a mixed-integer programming formulation (MILP), as well as faster alternatives, to compute SSE in this setting. We leverage the fact that while this setting is not zero-sum, it is so strategically as long as there are no ties among player utilities above, and the defender prefers that a target is covered, while the attacker prefers it to be uncovered. Consequently, the SSE can be computed by solving the following LP:

$$\min_{q} \quad v \tag{2.3.1a}$$

s.t.

$$\forall_t \quad v \geq q_t V_t^c + (1 - q_t) V_t^u \tag{2.3.1b}$$

$$\forall_t \quad \sum_t q_t = 1 \tag{2.3.1c}$$

$$\forall_t \quad q_t \geq 0. \tag{2.3.1d}$$

Characteristic to most SSE computation approaches using mathematical programming is that we compute the attacker's best response, and corresponding utility, using a set of constraints. Here, it is done using Constraints 2.3.1b. In the program above, the defender than simply minimizes the attacker's maximum utility; that is, he is computing a minimax strategy, which, it turns out, is equivalent to SSE in this highly restricted setting [66, 70].

The model above makes one particularly troublesome assumption: that we (the defender) know exactly what the attacker's preferences are. Typically, of course, we do not. The way to generalize this setting to allow uncertainty about attacker preferences is via Bayesian Stackelberg games, which are analogues to Bayesian games we discussed above. Specifically, suppose that there is a finite set of attacker types Θ, and let p_θ denote the probability that the attacker's type is θ. Let $V_t^u(\theta)$ and $V_t^c(\theta)$ denote the attacker's utilities when his type is $\theta \in \Theta$. Keeping a zero-sum framework for simplicity, we can rewrite the above LP as follows to compute a Bayesian Stackelberg equilibrium:

$$\min_{q} \quad \sum_\theta p_\theta v_\theta \tag{2.3.2a}$$

s.t.

$$\forall_{t,\theta} \quad v_\theta \geq q_t V_t^c(\theta) + (1 - q_t) V_t^u(\theta) \tag{2.3.2b}$$

$$\forall_t \quad \sum_t q_t = 1 \tag{2.3.2c}$$

$$\forall_t \quad q_t \geq 0. \tag{2.3.2d}$$

Chapter 3

Security for Interdependent Assets

3.1 Introduction

The revolution in communication and computing technologies has spurred unprecedented growth in connectivity, be it technical, economic, or social. Everyone benefits from an increasingly connected world: we can collect more information and make better decisions about the electric power grid by communicating with an increasingly complex network of sensors and smart devices, can lead a large-scale project with globally dispersed participants from the comfort of an office, and maintain active membership in a community, real or virtual, despite being geographically removed from its epicenter. The benefits are so patent, indeed, that associated risks are often easy to overlook. The risk of connectivity is that local failures can have global consequences. This is now well recognized in cybersecurity, as viruses propagate from system to connected system, often affecting a large fraction of businesses. Melissa virus, for example, affected more than 300 organizations, causing over $80 million in damage [34, 98]. As another example, the electric power grid, which is already a complex networks of generators, electric lines, along with businesses and households, is becoming much more so with the increasingly sophisticated sensors and "smart" meters, and with increasing complexity of the grid, security failures can have increasingly severe consequences [103, 96].

Despite the importance of accounting for interdependent risks in security decisions, there are few systematic approaches for empowering a decision maker to do so. The majority of the approaches to guide security investment decisions aspire to do so without explicitly accounting for interdependencies. For example, the standard approach to security risk management in the industry is to consider consequences in terms of asset value, consequence of a threat on that value, and frequency of threat, but either treats assets as independent, or abstracts away the complex interdependencies in a single cost/value measure [72]. Research in IT security management has largely been in line with this framework [113, 107, 32, 29, 43, 31, 88, 30, 33, 15]. The strands that explicitly model interdependent risk focus on spillover effects among many organizations or entities, rather than policy to secure interdependent assets [73, 89].

Our point of departure is a class of optimization-based game theoretic approaches in security settings referred to as *Stackelberg security games* [91]. These are two-player games in which a *defender* aims to protect a set of targets using a fixed set of limited defense resources, while an *attacker* aims to assail a target that maximizes his expected utility. A central assumption in the literature on Stackelberg security games is that the defender can commit to a probabilistic defense (equivalently, the attacker observes the probabilities with which each target is covered by the defender, but not the actual defense realization). Much of the work on Stackelberg security games focuses on building fast, scalable algorithms, often in restricted settings [66, 61, 101]. One important such restriction is to assume that targets exhibit *independence*: that is, the defender's utility only depends on which target is attacked and the security configuration at that target. Short of that restriction, one must, in principle, consider all possible combinations of security decisions jointly for all targets, making scalable computation elusive. Many important settings, however, exhibit interdependencies among potential targets of attack. These may be explicit, as in IT and supply chain network security, or implicit, as in defending critical infrastructure (where, for example, successful delivery of transportation services depends on a highly functional energy sector, and vice versa), or in securing complex software systems (with failures at some modules having potential to adversely affect other modules). While in such settings the assumption of independence seems superficially violated, we demonstrate below that under realistic assumptions about the nature of interdependencies, we can nevertheless leverage the highly scalable optimization techniques which assume independence.

In all, we offer the following contributions. First, we introduce a general framework to modeling security decisions for interdependent assets in the presence of both adversarial and non-adversarial threats. Second, we instantiate our general model of interdependencies using a graph in combination with an independent failure cascade model. Third, we present a general heuristic algorithm for computing approximately optimal security policies on networks that leverages submodularity of the attacker's problem in combination with a simple, yet highly effective, local search heuristic. Fourth, we present an important special case of our model which admits a highly scalable algorithm for computing optimal security policies *exactly*. Fifth, we apply our framework to study several applications of interdependent security, using both real networks, as well as stochastic generative network models. One of our most significant experimental contributions is an extensive study of comparative network resilience. This is a field which has had considerable significance in the broad network science literature and, indeed, is at the focus of two disparate strands of literature: the first comparing susceptibility of networks to attacks or random failures *when no defense is present*, and second, studying inoculation strategies on networks to protect from infectious disease spread, allowing for *no targeted attacks*. Our framework is the first that allows us to capture both endogenous defense measures and targeted attacks on networks, allowing us to unify these two strands of research. Our results, thus, provide much insight into both of these areas, offering additional nuance and, at times, contradicting the commonly held intuitions.

3.1.1 Literature Review

Our work is situated within the rapidly expanding body of literature on security investment and policy. Topically, this literature can be grouped into several streams. The first studies security policies from the perspective of liability considerations [30, 15], considering, for example, alternative ways to allocate burden or damage of security decisions (such as liability for zero-day exploits). The second is focused on the technical capabilities side, aiming to develop better intrusion detection systems (IDS), or IDS that are especially attuned to costs of decisions about classifying threats [95, 45, 74, 13]. The third stream, which is most closely connected to our aims, involves approaches to improve security investment decision support. Within this stream there are three general approaches: risk-based, decision theoretic, and game theoretic.

The risk-based approach is perhaps the oldest and seems still to be the principal approach in practice. At the crux of this approach is evaluation of specific security risks facing an organization, perhaps through an associated assessment of vulnerabilities, threats, and consequences [72, 42, 46, 81]. While much attention is paid in this literature on risk assessment and understanding threats (e.g., attackers), it offers relatively little quantitative guidance about *mitigation*, aside from the most basic cost-benefit comparison between deploying a particular security measure, and the expected risks and consequences it is meant to ameliorate.

The academic research community, in contrast, has aimed to shift focus on providing specific guidance about security investments, though in many cases this guidance is in very specific security contexts, such as whether or not to deploy a firewall or an IDS, and how to configure it if deployed [113, 107, 32, 29, 31, 33]. The corresponding approaches are either decision theoretic, modeling threats as unaffected by mitigation policies [113, 107, 32], or game theoretic, accounting for the impact of security policies on attackers' incentives [32, 29, 31, 33].

Game theoretic treatment of security is intimately connected to two simple classical models: inspection games [16] and colonel Blotto games [97]. The most basic variant of an inspection game involves an inspectee (e.g., a tax evader) who can choose to perform an illegal or a legal action, and an inspector, who receives a noisy signal upon which he can inspect (at some cost), or not. One qualitative difference between this generic inspector game and some of the models we described above, as well as our own approach, is that in our case the defender (inspector) acts first, and the attacker (inspectee) acts *after observing the defender's decision* (which may be randomized, in which case the attacker observes the probability distribution). Moreover, here, as in the above references, the defender's and attacker's action spaces are quite simple, and no interdependencies are relevant. Colonel Blotto game, too, is a simultaneous move game, but here two commanders are endowed with armies, and get to place a fraction of their force on each of n battlefields. Whichever side has the most forces on a battlefield wins that battle, and the winner of the game is the commander with the most battle victories. In this game, the decision space of each player is actually rather complex, though complex in a different way from our setting. However, the game is zero-sum (ours is not), and here again no interdependencies are typically modeled.

Insofar as interdependencies in security decisions have been modeled in related literature, this has been done in the context of interdependencies among multiple entities aiming to jointly defend their systems, with the focus on outcomes of strategic interactions, rather than offering a security policy for the *entire interdependent system* [73, 43, 89, 57]. For example, [73] study the problem of interdependencies among players, each deciding whether or not to invest in better security. There is no attacker in their model, so in that sense its scope is quite different from ours. Moreover, an individual player's decision is binary. What they aim to model are spillovers due to a decision not to secure one's own assets onto others, and they demonstrate that in many cases equilibrium exhibits insufficient security overall.

While most of the work described above considers either exogenously specified risks (e.g., natural disasters or human error), or deliberate attacks that adopt to the security policy, [115] were the first to consider both in a single comprehensive model as we do, albeit without explicitly modeling interdependent risks.

All the work described so far on game theoretic and decision theoretic approaches to security attempts to characterize decisions by the defender, attacker, or both using closed-form mathematical expressions. In parallel, there has been considerable literature that aspires to *compute* security decisions. One such stream involves numerous variants of *network interdiction* problems. At the high level, all such approaches start with a network flow or shortest path problem, with the goal of choosing an action (such as blocking a subset of nodes or arcs on the network) that most effectively reduces the flow or increases shortest paths [111, 41, 112, 24, 25, 82]. Like ours, these efforts all use mathematical programming formulations to compute an optimal interdiction strategy. Unlike our work, however, these efforts are fundamentally restricted to zero-sum games, account for interdependencies using models based on network flow, and in most cases do not include defense against interdiction, which is our focus here. [24] do present a tri-level formulation that attempts to allow one to take countermeasures against being interdicted by an attacker, but this model is extremely difficult to scale, making its practical utility quite limited.

Our point of departure is a class of optimization approaches for security decisions referred to commonly as *Stackelberg security games*. The paper that provided the computational foundations for what has become an active subfield of computational game theory was the work by Conitzer and Sandholm on computing optimal Stackelberg commitment strategies in general finite games [39]. In this paper, Conitzer and Sandholm presented the first algorithm for computing optimal randomized commitment strategies in Stackelberg games. [91] presented the first mixed-integer linear programming formulation for computing a Stackelberg equilibrium in Bayesian Stackelberg games. [66] introduce an important restricted class of Stackelberg games specifically targeted at security settings; they refer to these as *Stackelberg security games*, and demonstrate that extremely scalable algorithms can be devised for this class of games. Since then, a number of follow-up papers have emerged, studying, for the most part, computational aspects of the problem and aiming to scale the algorithms to larger and larger instances [75, 106, 109, 61, 69, 62, 38, 105], as well as illustrating their actual deployment in the field, such as the LAX airport [94], Federal Air Marshall Service [63], and the US Coast Guard [101]. Of these approaches, [105] presents the most similar model to ours. The principal difference is in the game structure and motivation: Tsai et al. model *both* the defender and attacker as agents who aim to influence contagion of ideas in a simultaneous move game; thus, the two players actually have symmetric roles. In our model, the attacker's goal is to start a failure cascade, but the defender aims to *minimize damages from cascading failures*, not start a cascade of his own.

3.2 Stackelberg Security Games

At the core of our model lies a *Stackelberg security game*, which consists of two players, the leader (defender) and the follower (attacker), and a set of possible targets. The leader can decide upon a randomized policy of defending the targets, possibly with limited defense resources. The follower (attacker) is assumed to observe the randomized policy of the leader, but not the realized defense actions. Upon observing the leader's strategy, the follower chooses a subset of targets to attack so as to maximize its expected utility. The typical solution concept for these games is a Strong Stackelberg Equilibrium (SSE), in which the leader plays an optimal policy that accounts for an follower's optimal response to it and, moreover, presumes that the follower breaks ties in the leader's favor.[1]

In past work, Stackelberg security game formulations focused on defense policies that were costless, but resource bounded, and security decisions amounted to covering (defending) a set of targets, or not. In numerous settings such models are quite limiting. For example, in cybersecurity, protecting computing nodes could involve configuring anti-virus and/or firewall settings, with stronger settings carrying a benefit of better protection, but at a cost of added inconvenience, lost productivity, as well as possible licensing costs. Indeed, costs on resources may usefully replace resource constraints, since such constraints are often not hard, but rather channel an implicit cost of adding further resources. Thus, our model allows the defender to choose among many *security configurations* for each valued asset, and, additionally, security resources are only available at some cost. Furthermore, while security games as described above naturally entail an attacker, in practice most failures are not at all a deliberate act of sabotage, but are due entirely to inadvertent errors. Thus, we also depart from previous literature on Stackelberg security games by explicitly modeling both attacks and random failures.

To formalize, suppose that the defender can choose from a finite set O of security configurations for each target $t \in T$, with $|T| = n$. A configuration $o \in O$ for target $t \in T$ incurs a cost $c_{o,t}$ to the defender. Let $s = \{o_1, \ldots, o_n\}$ be the (pure strategy) security configuration vector, with $o_t \in O$ denoting the security configuration chosen for target t; we refer to s as the *defense policy*. We denote by q_s the probability that the defender chooses a security configuration vector s. The attacker observes the randomized defense policy vector q, and chooses a subset of at most L targets to attack; let us denote this subset by $A = \{t_1, ..., t_L\}$. We denote the defender's utility function by $U(s,A)$ and the attacker's by $V(s,A)$ where s is the defense policy and A the attacker's response. To capture the distinction between active attacks and "nature", let r be the prior probability of the defender that a failure will happen due to a deliberate attack. If no attack is involved, any target can fail; the defender's belief that a set of targets B randomly fails (conditional on the event that no attack is involved) is g_B, with $\sum_B g_B = 1$.

[1] The idea that the follower breaks ties in the leader's favor may seem strange in the context of security games. However, note that the leader can make the follower strictly prefer the corresponding action by a slight change in his randomized policy.

3.3 Modeling Asset Interdependencies

3.3.1 A General Model

In this section we offer a general model of interdependencies among assets. We then present an important special case that admits a far more scalable approach for computing optimal security policies. Throughout this section we focus on the defender's utilities; attacker is treated identically.

Let w_t be an *intrinsic worth* of a target to the defender, that is, how much loss the defender would suffer if this target were to be compromised with no other target affected (i.e., not accounting for indirect effects). In doing so, we assume that these worths are independent for different targets. Moreover, suppose that when a target t is damaged or compromised (due to a successful attack either on t directly, or on another target which indirectly impacts t), only a fraction α_t of its worth remains. We allow α_t to be a random variable if the impact of an attack is non-deterministic. Let $z(A,s_A)$ be the probability that a subset A of targets fails (or are compromised) when these (and possibly other targets) are attacked and the defense configuration for nodes in A is s_A (that is, s_A is the portion of the defense vector s restricted to nodes in A). For example, if a failure of every node t due to an attack is independent of security configuration of other nodes, $z(A,s_A) = \prod_{t\in A} z(o,t)$, where $z(o,t)$ is the probability that node t fails if attacked when its security configuration is o. The defender utility when security configuration is s and the attacker attacks a subset A of targets is

$$U(s,A) = \sum_{\tilde{A}\subseteq A} z(\tilde{A},s_{\tilde{A}})E\left[\sum_{t'} \alpha_{t'}w_{t'} \mid s,\tilde{A}\right] = \sum_{\tilde{A}\subseteq A} z(\tilde{A},s_{\tilde{A}})\sum_{t'} w_{t'}E\left[\alpha_{t'} \mid s,\tilde{A}\right]. \qquad (3.3.1)$$

We can think of the term $E\left[\alpha_{t'} \mid s,\tilde{A}\right]$ as the expected damage to target t' when the subset of targets $\tilde{A}$ is successfully compromised by the attacker and the security configuration vector is s.

3.3.2 Cascading Failures Model

In general, one may use an arbitrary model to compute or estimate the consequences of node failures due to interdependence, $E\left[\alpha_{t'} \mid s,A\right]$. Here, we offer a specific model of interdependence between targets that is simple, natural, and applies across a wide variety of settings.

Let us fix the security policy vector s and the set A of targets that are initially compromised. Suppose that dependencies between targets are represented by a graph (T,E), with T the set of targets (nodes) as above, and E the set of edges (t,t'), where an edge from t to t' (or an undirected edge between them) means that target t' depends on target t and, thus, a successful attack on t may have an impact on t'. Each target has associated with it a worth, w_t, as above, although in the current context this worth is incurred only if t is affected (e.g., compromised, broken). We model the interdependencies between the nodes as independent cascade contagion, which has previously been used primarily to model diffusion of product adoption and infectious diseases [65, 44]. The contagion proceeds starting at the attacked nodes $t \in A$, affecting each of their network neighbors t' with probability $p_{t,t'}(s)$, then spreads from each affected t', and so on, recursively. Contagion can only spread once along any network edge, and if a node is affected, it remains affected through the diffusion process (note also that in this model, the node is either affected, or not; we let $\alpha_t = 0$ when a node is affected by an attack and $\alpha_t = 1$ when it is not[2]). An equivalent way to model this process is to start with the network (T,E) and remove each edge (t,t') with probability $(1 - p_{t,t'}(s))$. The entire connected component of each attacked node is then deemed affected. As an important special case, we can use $p_{t,t'}(o_{t'})$ to model the impact of inoculation on the probability of becoming infected, for example, setting it to 0 if $o_{t'}$ is the decision to administer inoculation on node t' and to 1 if $o_{t'}$ is the decision not to inoculate t'.

3.3.3 Computing Expected Utilities

In principle, our setup allows us to fully decouple computing or estimating expected utilities $U(s,A)$ and $V(s,A)$ of the defender and the attacker respectively, and subsequently computing an optimal defense policy. In general, we can estimate player utilities by simulating cascades starting at every subset of nodes $\tilde{A}$ of size at most L and for every (deterministic) security configuration vector s, with expected utility of defender/attacker estimated as a sample average over K simulated cascades to obtain estimates of $E\left[\alpha_{t'} \middle| s,\tilde{A}\right]$, and applying Equation 3.3.1. Clearly, however, even estimating expected utilities for the entire game is an entirely intractable process in our general setup. Consequently, in the fully general case, we would wish to compute or approximate a Stackelberg equilibrium without having to know the full payoff functions of both players. Below, we demonstrate how this can be done using a combination of heuristic and submodular optimization methods. For the moment, however, we introduce a special case which allows us to compute an optimal security policy exactly and efficiently.

3.3.4 Special Case: Single-Node Attacks and Security-Independent Cascades

The most basic problem with the general setup that we described above is that in order to estimate the defender and attacker utility functions, and ultimately compute optimal security strategies, one needs to perform a set of simulations *for each defense policy vector s and attack strategy A*. Clearly, this becomes intractable even for a modest number of targets. In this section, we introduce several restrictions on the general model that allow both a much more compact representation of the players' payoff functions, and, ultimately, offers an opportunity for highly scalable Stackelberg equilibrium computation.

The first restriction is that the attacker can only attack a single target. Note that under this restriction, Equation 3.3.1 simplifies to

$$U(s,t) = z(o,t) \sum_{t'} w_{t'} E\left[\alpha_{t'} \middle| s,t\right]. \tag{3.3.2}$$

Indeed, this restriction has been operational in most related work on computing strong Stackelberg equilibria in the context of security [66, 61, 101]. The second restriction is captured by the

[2] Note that it is direct to replace these choices by arbitrary different constants

following condition on the impact of interdependencies:

Condition 3.3.1. *For all t and t', $E\left[\alpha_{t'} \mid s,t\right] = E\left[\alpha_{t'} \mid o_t,t\right]$.*

In words, the probability that a target t' is affected when an initially attacked target t fails only depends on the security configuration at the attacked target t. Below, we use o instead of o_t where t is clear from context.

There are several natural ways to think about Condition 3.3.1. The simplest is consider the consequences of attacks as affecting network flows. In this case, removing a node t and its incident edges from a network means that any flow between a pair of other nodes s,r must take a different route and, indeed, it may even be that s and r are now disconnected. Significantly, the utility lost in this case only depends on the security configuration at t. An alternative way to interpret Condition 3.3.1 is that security against external threats is not very efficacious once an attacker has found a way into the system. For example, in cybersecurity defense is often focused on external threats, with little attention paid to threats coming from computers internal to the network. Thus, once a computer on a network is compromised, the attacker may find it much easier to compromise others on the same network. This second interpretation gives rise to a very natural restriction on the cascading failure model that satisfies Condition 3.3.1: $p_{t,t'}$ do not depend on security configurations at nodes. This restriction is very common, as argued above. There is, however, an important setting in which it is clearly unrealistic: bioterrorism, where inoculation decisions reduce the likelihood of an individual being infected either by the attacker, or by another infected individual.

Under Condition 3.3.1, the defender's utility when t is attacked under security configuration o becomes:

$$U(o,t) = z(o,t)w_t E[\alpha_t|o_t,t] + \sum_{t' \neq t} w_{t'} E[\alpha_{t'}|o_t,t].$$

Thus, in this special case, we can represent the game much more compactly, using $U(o,t)$ and $V(o,t)$ to denote the defender's and attacker's utility, respectively, when target t is attacked and the security configuration at that target is o. In a slight abuse of notation, we denote by $q_{o,t}$ the probability that the defender chooses o at target t. Note that given q_s, we can compute $q_{o,t}$ as $q_{o,t} = \sum_s q_s 1(s_t = o)$, where $1(\cdot)$ is an indicator function which is 1 when its argument is true and 0 otherwise. Capturing the natural disasters in this special setting requires us (for algorithmic reasons) to restrict nature to affect a single target at a time. Thus, we will abuse notation again, denoting by g_t the probability that target t randomly fails (conditional on the event that no attack is involved), with $\sum_t g_t = 1$.

Observe that even when Condition 3.3.1 is operational, if we use the independent cascades model for failures above, we still need to estimate the consequence of cascades starting from each node t (i.e., from each target of possible attack). In several special cases, however, we can either compute player expected utilities from cascading failures exactly and efficiently, or substantially speed up utility estimation. We now address these special cases.

Cascades on Trees

It is intuitive that when the dependency graph is a tree, expected utilities can be computed efficiently. A naive algorithm can do it in linear time *for each target t*, yielding quadratic time in total (since we must repeat the process for all targets). In fact, we can do it in linear time *for all targets*, as the following theorem asserts.

Theorem 3.3.1. *Suppose the attacker can only attack a single target and condition 3.3.1 holds. If* (T,E) *is an undirected tree we can compute expected utilities for at targets in* $O(|T|)$ *time.*

Proof. Let us define the neighbors of a target t as N_t. By definition, the expected utility of a given node t ($U(o,t)$) is the direct utility at that node ($z(o,t)w_t$) plus the expected utility due to the cascading failure. The expected utility due to cascading failure is

$$z(o,t)\sum_{t'\neq t} w_{t'}p(failure(t')|t)$$

where $p(failure(t')|t)$ is the probability a node t' fails if node t fails. Since this is a tree, there is only one path between any pair of nodes, which means we can express $p(failure(t')|t)$ as the product of probabilities of the edges on the path between t and t'.

Next, let us consider the set of paths generated by each pair of nodes in the tree. If we organize these paths by the edges they contain (and use linearity of expectation), we can express the expected utility of the contagion spreading across an edge (t,t'), $E[U_{(t,t')}]$, as:

$$E[U_{(t,t')}] = p_{t,t'}\left(w_{t'} + \sum_{t''\in N_{t'},t''\neq t} E[U_{(t',t'')}]\right). \tag{3.3.3}$$

Thus, we can reason that for each node t:

$$U(o,t) = z(o,t)U_t,$$

where

$$U_t = w_t + \sum_{t'\in N_t} E[U_{(t,t')}] \tag{3.3.4}$$

Now let us describe a two-pass algorithm for calculating U_t for all t. First, choose an arbitrary node to be the root of the tree. In the first pass, we calculate the expected loss due to each edge from parent to child ($E[U_{(P,C)}]$) from the bottom of the tree upward. In the second pass, we calculate the expected loss on each edge from child to parent ($E[U_{(C,P)}]$) from the top of the tree downward. We can model this as a message passing algorithm, where calculating $E[U_{(t,t')}]$ is done by passing a message from t' to t. We can see by Equation 3.3.3 that the necessary inputs to calculate $E[U_{(t,t')}]$ are the messages from $N_{t'}\setminus t$ to t'. We will now show that at the time that each of these messages is generated, all of the necessary inputs will be available.

Consider the edges between a given node t and its neighbors. Unless t is the root, one of these edges will be between t and its parent P_t, and the rest (possibly 0 in the case where t is a leaf node) will be between t and its children. Since in the first pass we are passing messages from child to parent and a node has only one parent, we will have received messages from $N_t \setminus P_t$ when we generate the message from t to P_t.

For the second pass, when we pass information to a child of t, C_t, we will have received messages from N_t, thus we again have the necessary information to generate the message from t to C_t.

Finally, once a node has received messages from all of its neighbors we can easily calculate the expected loss at each node by Equation 3.3.4. However, to achieve a runtime of $O(n)$, we need to be slightly more clever in how we store these values. By combining Equations 3.3.3 and 3.3.4 we can reason that $E[U_{(t,t')}] = p_{t,t'}(U_{t'} - E[U_{(t',t)}])$. This allows us to give an equivalent definition of U_t:

$$U_t = w_t + \sum_{t' \in N_t} p_{t,t'}(U_{t'} - E[U_{(t',t)}]). \tag{3.3.5}$$

Now, consider the same two-pass algorithm as before, but rather than storing the expected loss for every edge, we merely store a running total of the expected loss at each node. We argue that by the same reasoning as before that the necessary calculations will have been performed before we need them as inputs. However, we still need to show that we can recover the correct value out of the values stored at the two nodes. When we calculate $E[U_{(P,C)}]$ in the first pass, the value stored at C will be $(U_C - E[U_{(C,P)}])$, since we have not yet updated C with $E[U_{(C,P)}]$. However, when we reach this edge on the downward pass to to calculate $E[U_{(C,P)}]$, P will have U_P stored. Since the value stored at C is still $(U_C - E[U_{(C,P)}])$, we can easily calculate $E[U_{(C,P)}] = p_{C,P}(U_P - E[U_{(P,C)}]) = p_{C,P}(U_P - p_{P,C}(U_C - E[U_{(C,P)}]))$ and update C.

Since we visit each edge twice, and perform a constant amount of work each time, we can bound the runtime by $O(|E|)$. Since in a tree $|T| - 1 = |E|$, we can also bound the runtime by $O(|T|)$. □

Cascades on Undirected Graphs

In general undirected graphs, we can apply a very simple optimization in the way we sample cascades to obtain substantial speedups when the graph is dense. First, observe that rather than determining live edges as the cascade unfolds, we can instead flip the biased coin for each edge to determine whether it is live or not during a particular cascade *prior* to propagating the failure. The resulting graph contains a subset of edges from the original graph. At this point, observe that each potential target in a given connected component will result in the same defender/attacker utility. We therefore only need to compute the expected loss once for each connected component. When the size of the largest connected component is $O(|T|)$, a likely scenario in dense graphs, this optimization results in an $O(|T|)$ speedup.

3.3.5 The Significance of Capturing Interdependence

An obvious question that may arise upon pondering the complexities of our framework is whether they are worthwhile: it may well be that previous approaches which assume target independence offer satisfactory approximation. We now show theoretically, and later experimentally, that our approach improves dramatically as compared to one which assumes independence.

Proposition 3.3.2. *There exists a family of problem instances for which the independence assumption yields a solution that is a factor of $O(n)$ worse than optimal.*

Proof. Consider a star with edges all directed towards the spokes and cascade probabilities of 1. All nodes have worth $w_t = 1$ for both defender and attacker. Now, suppose that cost of defending a target is $c = 2$. If targets are independent, it is clearly not worth defending any of them. But since they are, in fact, dependent, and the attacker will attack the hub, the utility of the defender is $-1 * n = -n$. If the defender recognizes the dependencies, he will protect the hub, and the total loss will be -3. □

Additionally, we now show that allowing for interdependence, but assuming that cascades are independent of security decisions, also comes at a substantial loss.

Proposition 3.3.3. *There exists a family of problem instances for which assuming that cascades are independent of security decisions yields a solution that is a factor of $O(n)$ worse than optimal.*

Proof. Consider again a star, now undirected, and cascade probabilities still 1. Suppose that the hub has worth $w_t = n$ and the rest of the nodes have no worth, and the cost of defense is $c = 1$. If we assume that defending the hub will still leave it susceptible, the defender must defend all nodes, or none at all. Defense in this case will cost $n+1$, and since the benefit is only n, the defender will not protect anything, yielding a utility of $-n$. However, since in actuality it suffices only to protect the hub, the utility of an optimal defense is -1. □

3.3.6 Incorporating Uncertainty about the Network

Applying our framework in real-world networked security settings requires an accurate understanding of the interdependencies. Thus far, we assumed that the actual network over which cascading failures would spread is perfectly known. A natural question is: what if our network model is inaccurate?

Formally, we model the uncertainty about the network as a parameter ε which represents the probability of incorrectly estimating the relationship between a pair of targets. Thus, if there is an edge between t and t', we now let this edge be present with probability $1-\varepsilon$. On the other hand, if t and t' are not connected in the graph given to us, we propose that they are, in fact, connected with probability ε. Thus, when the graph is large, even a small amount noise will cause us to err about a substantial number of edges.[3]

Note that there is a natural way to incorporate this model of uncertainty into our framework. Let us interpret $p_{t,t'}(o_t, o_{t'})$ as the probability of a cascade from t to t' *conditional on an edge from t to t'*. Then, if t and t' are connected, we modify cascade probabilities to be $\hat{p}_{t,t'}(o_t, o_{t'}) = p_{t,t'}(o_t, o_{t'})(1-\varepsilon)$, whereas if they are not connected, the cascade probability is $\hat{p}_{t,t'}(o_t, o_{t'}) = p_{t,t'}(o_t, o_{t'})\varepsilon$.

3.4 Computing Optimal Randomized Security Configurations

3.4.1 The General Case: Exact Solution

Previous formulations of Stackelberg games for security involved a fixed collection of defender resources, and in most cases a binary decision to be made for each target: to cover it, or not. To adapt these to our domains of interest, we first modify the well-known multiple linear program (henceforth, multiple-LP) formulation to incorporate an arbitrary set of security configurations, together with their corresponding costs of deployment. In the multiple-LP formulation, each linear program solves for an optimal randomized defense strategy *given that the attacker attacks a fixed subset of targets* $\hat{A}$, with the constraint that $\hat{A}$ is an optimal choice for the attacker. The defender then chooses the best solution from all feasible LPs as his optimal randomized defense configuration. The LP formulation for a representative subset of targets $\hat{A}$ is shown in Equations 3.4.1a-3.4.1d.

$$\max \quad r\left(\sum_s U(s,\hat{A})q_s^{\hat{A}}\right) + (1-r)\left(\sum_{B,s} g_B U(s,B) q_s^{\hat{A}}\right) - \sum_t \sum_o c_{o,t} q_{o,t}^{\hat{A}}. \tag{3.4.1a}$$

s.t.

$$\forall_s \quad q_s^{\hat{A}} \in [0,1] \tag{3.4.1b}$$

$$\sum_s q_s^{\hat{A}} = 1 \tag{3.4.1c}$$

$$\forall_A \quad \sum_s V(s,A) q_s^{\hat{A}} \le \sum_s V(s,\hat{A}) q_s^{\hat{A}} \tag{3.4.1d}$$

The intuition behind the multiple-LP formulation is that in an optimal defense configuration, the attacker must (weakly) prefer to attack *some* subset of targets, and, consequently, one of these LPs must correspond to an optimal defense policy.

[3] We assume here that both the defender and attacker share the same uncertainty about the network. An alternative model could consider an attacker that has more (or exact) information about the network. The resulting defender problem would become a Bayesian Stackelberg game.

3.4.2 Approximating Security Policy in the General Case

There are two significant problems with the LP formulation for computing optimal defender policies we described above. First, the LP itself becomes intractably large when we have a sufficient number of network nodes and defense configuration options. Perhaps a far more significant problem, however, is that the LP requires us to first compute or estimate the expected utilities for each joint strategy of the defender and attacker based on our model of interdependencies. It is this bottleneck, as much as any other, that renders the exact approach intractable in practice.

In this section, we offer an alternative that takes advantage of the special structure in the independent failure cascades model. This alternative approach allows us to avoid estimating the entire payoff matrix, interleaving optimization and estimation steps instead in a manner analogous to simulation-based game theoretic analysis [110]. To simplify the problem, we restrict attention here to deterministic defense policies; generalization is immediate if we discretize randomized policies.

We begin by focusing on the attacker's best response problem, an algorithmic challenge in its own right, and subsequently proceed to propose a local search heuristic to obtain a defender's policy in which the attacker's optimization problem is a subroutine. We assume henceforth that the interdependencies among the targets are modeled using the dependency graph and independent failure cascades.

Approximating an Optimal Attack

The attacker's problem is to choose a subset of L targets to attack so as to maximize his expected utility $V(s,A)$. This problem is a generalization of the well-known problem of influence maximization [65], in which a decision maker aims to maximize the expected number of individuals (rather than utility) affected by a cascade started from the chosen nodes. Kempe et al. showed that the problem of choosing an optimal subset of L nodes to seed when subsequent influence spreads according to an independent cascades model is NP-Hard. In our setting, the attacker's problem is a slight generalization of this model, and NP-Hardness of the attacker's problem is therefore immediate (setting $w_t = 1$ for all nodes recovers the original influence maximization problem).

Theorem 3.4.1. *Computing an optimal attack strategy is NP-Hard.*

An important algorithmic insight by Kempe et al. is that while solving the influence maximization problem optimally is hard, the objective function is *submodular*. Consequently, a simple greedy heuristic yields a constant factor approximation and, in practice, gives nearly optimal solutions. While our setting is slightly more general, we can readily extend this submodularity result.

Theorem 3.4.2. *The attacker's objective function is submodular.*

Proof. Note that the cascade process can be equivalently formulated by first flipping the biased coins for each edge, keeping the edge between t and t' with probability $p_{t,t'}(o_t, o_{t'})$ and deleting it otherwise. The total utility to the attacker given such a realization is the sum of the worths of all targets affected by the attacker's decision A. Let T_t be the set of targets with a finite path from a particular target t, and let $T_R = \cup_{r \in R} T_r$ be the set of targets reachable from any target in a set R. Finally, for any set of targets $R \subseteq T$, define $U(R) = \sum_{r \in R} w_r$, that is, the total worth of all targets in R.

Suppose $R \subseteq S \subseteq T$ be targets of initial attack and consider attacking an additional target t'. The attacker's utility when the set R of targets is attacked is $U(T_R)$, while the utility from attacking targets in $R \cup t'$ is $U(T_{R \cup t'})$. Then,

$$U(T_{R\cup t'}) - U(T_R) = \sum_{r \in T_{R\cup t'}} w_r - \sum_{r \in T_R} w_r = \sum_{r \in T_{R\cup t'} - T_R} w_r.$$

Now, observe that if $R \subseteq S$, $T_{S\cup t'} - T_S \subseteq T_{R\cup t'} - T_R$, which implies that

$$\sum_{r \in T_{R\cup t'} - T_R} w_r \geq \sum_{r \in T_{S\cup t'} - T_S} w_r = U(T_{S\cup t'}) - U(T_S),$$

which in turn implies that for every realization of the random cascade graph, the attacker utility is submodular. Since submodularity is preserved under linear transformations, the attacker expected utility is also submodular. □

The implication is that for a fixed defense policy s we can approximate the optimal attack to a factor of $1 - 1/e$ with an iterative greedy algorithm which chooses, in each iteration, the target to attack that attains the highest increase in expected utility with respect to previously chosen targets [83].

Computing a Defense Policy

Thus far we have shown that we can compute a near-optimal strategy for the attacker reasonably fast. We now come to the main problem: computing a defense policy. First, we observe that while the attacker's problem is submodular, this is not the case for the defender: defense decisions have complementarities. These arise because targets are interdependent and, therefore, defending one target may have little effect until other targets connected to it are also defended. The presence of such complementarities would in principle make the combinatorial optimization problem faced by the defender extremely difficult. However, we offer a simple local search heuristic and show empirically that it is highly effective, particularly when combined with random restarts.

To begin, let us make several basic structural observations. First, if a particular security configuration o is less effective than another, o', and is at the same time more expensive than o', we can prune it from consideration, since it is *dominated* by o', a notion which we now formally define.

Definition 3.4.1. A security configuration o' is *stronger* than o if $z(o,t) \geq z(o',t)$ for all $t \in T$, $p_{t,t'}(o',o') \geq p_{t,t'}(o',o)$, $p_{t,t'}(o',o') \geq p_{t,t'}(o,o')$, and $p_{t,t'}(o',o') \geq p_{t,t'}(o,o)$ for all $t,t' \in T$.

Definition 3.4.2. A security configuration o is *dominated* if $\exists o' \in O$ with $c_{o',t} \leq c_{o,t}\ \forall\, t \in T$ that is stronger than o (i.e., o' is both stronger and cheaper).

Second, suppose that cascade probabilities do not depend on security configurations (a special case of our model). In this case, increasing the amount of defense (formally, choosing a stronger security configuration that is more expensive) at a particular target has no value to the defender unless either this target is attacked, or the defender simultaneously increases defense at another target that is. The reason is that since the attacker's decision is not affected, the only consequence is the increased cost to the defender. While this observation is no longer true when cascade probabilities depend on defense, we nevertheless base our local search on it, and view it as a heuristic in the general case.

We propose a simple local search algorithm (Algorithm 1) that iteratively chooses a single target at a time, distinguishing between those that are currently attacked and those that are not based on the second observation above, and chooses a locally optimal security configuration for that target.

```
Data: Starting defense policy $s_0$, number of iterations $I$
Result: Final defense policy $s$
$s \leftarrow s_0$;
prune all dominated $o \in O$;
for $i = 1$ to $I$ do
    $A \leftarrow computeAttack(s)$ // targets attacked under $s$ for $t \in A$ do
        // fix all other decisions
        // compute the local optimum at target $t$
        $o_t \leftarrow computeBest(t)$;
        $s \leftarrow \{s_1, \ldots, o_t, \ldots, s_n\}$;
    end
    for $t \notin A$ do
        // compute local optimum, considering only decreasing security
        $o_t \leftarrow computeBestDecrease(t)$;
        $s \leftarrow \{s_1, \ldots, o_t, \ldots, s_n\}$;
    end
end
```

Algorithm 1: Local search for a defense policy.

Algorithm 1 requires as input an initial defense policy from which to start local search. Two natural candidates are the weakest and strongest policies, i.e., a policy in which every target is using a weakest (resp. strongest) security configuration, if these exist. As an example, one usually has an option of “no security”, which is the weakest option, and “high security”, which would be the strongest. A third natural candidate is a well-known heuristic, choosing individuals to defend in decreasing order of degree; this is commonly referred to as *targeted vaccination* [93]; since this heuristic plays an important role in the literature on vaccination on networks, below we show experimentally that in isolation it is significantly worse than our local search method. Finally, we can start from a random defense policy. Ultimately, since this is only a local search, and our problem exhibits complementarities, we would not expect it to yield optimal solutions in general. Therefore, our full approach runs the local search from the weakest and strongest defense policy, if these exist, then from a configuration based on targeted vaccination, and finally runs it from

P random starting policies. Below, we show empirically that the local search often yields nearly optimal solutions even without random restarts.

Note that local search implicitly invokes a subroutine for computing an optimal attacker strategy; this is actually explicit in the *computeAttack*(s) function call and implicit in both functions computing locally best security configuration at a given target. If we could compute this strategy optimally, we could guarantee that our overall approach converges to an optimal defense with probability 1 if we let the number of random restarts grow without bound. While this is easy to guarantee when the attacker can only attack a single target, it is no longer reasonable when the attacks can happen on multiple targets simultaneously. Nevertheless, if the game is nearly constant-sum (in the sense we formalize presently), computing an approximately optimal attacker strategy suffices to guarantee convergence to an approximately optimal defense. For convenience, suppose that both the attacker and defender always obtain non-negative payoffs.

Definition 3.4.3. A security game is ε-constant-sum if there exists $c \geq 0$ such that $c - \varepsilon \leq U_{s,a} + V_{s,a} \leq c + \varepsilon$ for all s, a.

Theorem 3.4.3. *Suppose that the game is ε-constant-sum. Additionally, suppose that $\hat{A}(s)$ is an α-approximation of an optimal attacker strategy $A^*(s)$ for a given defense policy s. Let $\hat{s}$ be an optimal defender policy if the attacker response is measured according to $\hat{A}$, and let s^* be the true optimal policy. Then $U(\hat{s}, A^*(\hat{s})) \geq U(s^*, A^*(\hat{s})) - (\alpha - 1)V(s^*, A^*(\hat{s})) - 2\varepsilon(\alpha + 1)$.*

Proof. Choose an arbitrary defence policy s. Since $\hat{A}(s)$ is an α-approximation (for $\alpha \geq 1$),

$$\alpha V(s, \hat{A}(s)) \geq V(s, A^*(s)).$$

Using $c - \varepsilon \leq U(s,A) + V(s,A) \leq c + \varepsilon$ for all s, A, this implies that

$$\alpha(c - U(s, \hat{A}(s)) + \varepsilon) \geq c - U(s, A^*(s)) - \varepsilon,$$

or, equivalently,

$$U(s, A^*(s)) \geq \alpha U(s, \hat{A}(s)) - c(\alpha - 1) - \varepsilon(\alpha + 1).$$

Since this is true for every s,

$$\begin{aligned} U(\hat{s}, A^*(\hat{s})) &\geq \alpha U(\hat{s}, \hat{A}(\hat{s})) - c(\alpha - 1) - \varepsilon(\alpha + 1) && (3.4.2) \\ &\geq \alpha U(s^*, \hat{A}(s^*)) - c(\alpha - 1) - \varepsilon(\alpha + 1) && (3.4.3) \\ &\geq \alpha U(s^*, A^*(s^*)) - c(\alpha - 1) - \varepsilon(\alpha + 1) - 2\varepsilon && (3.4.4) \\ &= \alpha U(s^*, A^*(s^*)) - c(\alpha - 1) - \varepsilon(\alpha + 3), && (3.4.5) \end{aligned}$$

where inequality 3.4.3 follows because of optimality of $\hat{s}$ for the defender under $\hat{a}$ and inequality 3.4.4 is due to the fact that $\hat{a}$ is suboptimal for the attacker. Rearranging and letting $c \leq U(s^*, A^*(s^*)) + V(s^*, A^*(s^*)) + \varepsilon$ we get the desired result. □

If $V(s^*, A^*(s^*))$ is relatively small (e.g., attacker gains are a relatively small fraction of available value) and $\alpha - 1 \approx 0.6$ (as is the case when we use the greedy algorithm to approximate attacker's policy), we can be sure to be relatively close to optimal defender utility with sufficiently many random restarts and sampled cascades.

3.4.3 Special Case: Single-Node Attacks and Security-Independent Cascades

The multiple-LP formulation 3.4.1 for the general case requires us to have a variable for each possible security configuration vector *and* requires us to solve an LP for each subset of L targets. Since the number of possible configurations, as well as the number of possible subsets of targets, is exponential in the number of targets, exact security policy computation cannot scale beyond very small instances. However, if we assume that the attacker can attack at most a single target, restrict random failures to a single target at a time, and assume that the defender's utility only depends on the target being attacked or failing (Condition 3.3.1), we can obtain a far more compact and scalable formulation. Under these assumptions, we can treat the defense configuration for each target $q_{o,t}$ in isolation, as we no longer need to randomize over joint defense schedules. Moreover, we need only solve n LPs, one for each target $\hat{t}$ of possible attack. The LP formulation for a representative target $\hat{t}$ is shown in Equations 3.4.6a-3.4.6d.

$$\max \quad r\left(\sum_{o} U(o,\hat{t})q^{\hat{t}}_{o,\hat{t}}\right) + (1-r)\left(\sum_{t,o} g_t U(o,t)q^{\hat{t}}_{o,t}\right) - \sum_{t}\sum_{o} c_{o,t}q^{\hat{t}}_{o,t}. \tag{3.4.6a}$$

s.t.

$$\forall_{o,t}\ q^{\hat{t}}_{o,t} \in [0,1] \tag{3.4.6b}$$

$$\forall_{t} \sum_{o} q^{\hat{t}}_{o,t} = 1 \tag{3.4.6c}$$

$$\forall_{t} \sum_{o} V(o,t)q^{\hat{t}}_{o,t} \leq \sum_{o} V(o,\hat{t})q^{\hat{t}}_{o,\hat{t}} \tag{3.4.6d}$$

Notice that we can easily incorporate additional linear constraints. For example, it is often useful to add a budget constraint of the form:

$$\forall_{\hat{t},t} \quad \sum_{o} c_{o,t}q^{\hat{t}}_{o,t} \leq C.$$

The Impact of Sampling Noise

While we can compute the expected utilities exactly in certain important special cases (see [78]), in general we must sample cascades to estimate expected utilities of players, and solve the optimization problem (3.4.6a-3.4.6d) using estimated utilities. This raises a natural question: does this approach yield a solution close to optimal if we take sufficient samples of cascades, and thereby obtain an arbitrarily good estimate of utilities for all outcomes? The answer, it turns out, is non-trivial, because sampling noise does not merely affect the objective functions of the LPs we solve, but also the constraints.

To appreciate what can go wrong, consider an example with two targets, 1 and 2, and suppose that there are only two security configurations: a target can either be covered or not. Let U^u_t and U^c_t be the defender's actual utilities if target t is uncovered and covered, respectively, and, similarly, let V^u_t and V^c_t be the corresponding utilities for the attacker, and let $r = 1$. Moreover, suppose that $V_1^u = V_1^c = V_2^u = V_2^c = 1$, that is, the attacker is completely indifferent between the targets and defender strategy choices. Assume that $U_1^u = -K$, and $U_1^c = U_2^u = U_2^c = 0$. That is, the defender prefers that the attacker attacks target 2. Finally, let the cost of leaving a target uncovered be 0, and coverage costs be $c_1 = c_2 = K/2$. Clearly, the optimal defender strategy is to cover nothing, because the attacker's indifference will result in him attacking target 2 in a strong Stackelberg equilibrium.

Now, suppose that we add some mean-zero random noise to the attacker's payoffs. With probability $1/24$, the attacker's payoffs will be perceived to be ordered as follows: $\hat{V}_2^c < \hat{V}_2^u < \hat{V}_1^c < \hat{V}_1^u$. This ordering implies that the attacker will prefer to attack target 1 *no matter what the defender's strategy is*. Thus, the LP for target 2 will be infeasible, and the LP for target 1 is always feasible. The objective value of the LP for target 1 can be written as

$$\max_{q_1,q_2} \frac{K}{2}q_1 - \frac{K}{2}q_2,$$

where q_1 and q_2 are the probabilities of covering targets 1 and 2 respectively. Clearly, the optimal solution is to have $q_1 = 1$ and $q_2 = 0$, yielding an actual loss to the defender of $K/2$ (due to unnecessary security expenditures), compared to 0 in an optimal solution.

We now show that if we restrict the game to be strictly competitive, we do indeed obtain convergence to an optimal solution if we increase the number of samples. Let O^* be the true optimal utility of the defender (when the utilities are computed exactly), define $\hat{q}$ as an optimal solution when the player utilities are computed from samples, and let $O(\hat{q})$ denote the actual defender utility when the security policy is $\hat{q}$. Let $\hat{U}(o,t)$ denote the estimate of the defender's utility function.

Theorem 3.4.4. *Suppose that the game is strictly competitive and suppose that $|\hat{U}(o,t) - U(o,t)| \leq \varepsilon$ for all o,t. Then $O(\hat{q}) \geq O^* - 2\varepsilon$.*

Proof. When the game is zero-sum, an optimal solution can be computed using the following simpler, single-LP formulation:

$$\max \quad r\left(\min_t \sum_o U(o,t)q_{o,t}\right) + (1-r)\left(\sum_{t,o} g_t U(o,t)q_{o,t}\right) - \sum_t \sum_o c_{o,t}q_{o,t} \tag{3.4.7a}$$

s.t.

$$\forall_{o,t}\ q_{o,t} \in [0,1] \tag{3.4.7b}$$

$$\forall_t \sum_o q_{o,t} = 1. \tag{3.4.7c}$$

First, note that the solution $\hat{q}$ obtained when utilities are estimated is feasible for program 3.4.7

where actual utilities are used. Thus, we can focus just on the objective value. Then,

$$
\begin{aligned}
O(\hat{q}) &= r\left(\min_t \sum_o U(o,t)\hat{q}_{o,t}\right) + (1-r)\left(\sum_{t,o} g_t U(o,t)\hat{q}_{o,t}\right) - \sum_t \sum_o c_{o,t}\hat{q}_{o,t} \\
&\geq r\left(\min_t \sum_o \hat{U}(o,t)\hat{q}_{o,t}\right) + (1-r)\left(\sum_{t,o} g_t \hat{U}(o,t)\hat{q}_{o,t}\right) - \sum_t \sum_o c_{o,t}\hat{q}_{o,t} - \varepsilon \\
&\geq r\left(\min_t \sum_o \hat{U}(o,t)q^*_{o,t}\right) + (1-r)\left(\sum_{t,o} g_t \hat{U}(o,t)q^*_{o,t}\right) - \sum_t \sum_o c_{o,t}q^*_{o,t} - \varepsilon \\
&\geq r\left(\min_t \sum_o U(o,t)q^*_{o,t}\right) + (1-r)\left(\sum_{t,o} g_t U(o,t)q^*_{o,t}\right) - \sum_t \sum_o c_{o,t}q^*_{o,t} - 2\varepsilon \\
&= O^* - 2\varepsilon.
\end{aligned}
$$

□

Since the number of security configurations o and targets t is finite, we can obtain the uniform bound required by Theorem 3.4.4 directly from the law of large numbers. Thus, the theorem implies that as we take more samples, the resulting solutions converge to optimal in terms of the defender's utility.

3.5 Illustrations

In this section we illustrate our framework on two simple examples. The first is an artificial supply chain example that we constructed. The second uses a graph of interdependencies among critical infrastructure and key resource sectors obtained from the DHS and FEMA websites. For both these examples, we use the exact approach in the restricted setting with an attacker only attacking a single node and cascades that do not depend on security decisions.

3.5.1 A Simple Supply Chain

Consider a seven-node supply chain (directed acyclic graph) shown in Figure 3.1. We suppose that the entire supply chain (or at least the relevant security decisions) is controlled by a single firm which is primarily concerned with manufacturing two types of cars, one more profitable than the other. The actual components that ultimately comprise the cars are not intrinsically valuable to the manufacturer (or are valued so low relative to the final product as to make them effectively unimportant in this decision). All parts of the supply chain may be inspected at some cost c, or not (in which case no cost is incurred).

The first step in our framework is to compute (or estimate) the expected utility for each node in the supply chain. To do this, we first specify the probability that an attacked node is affected (in

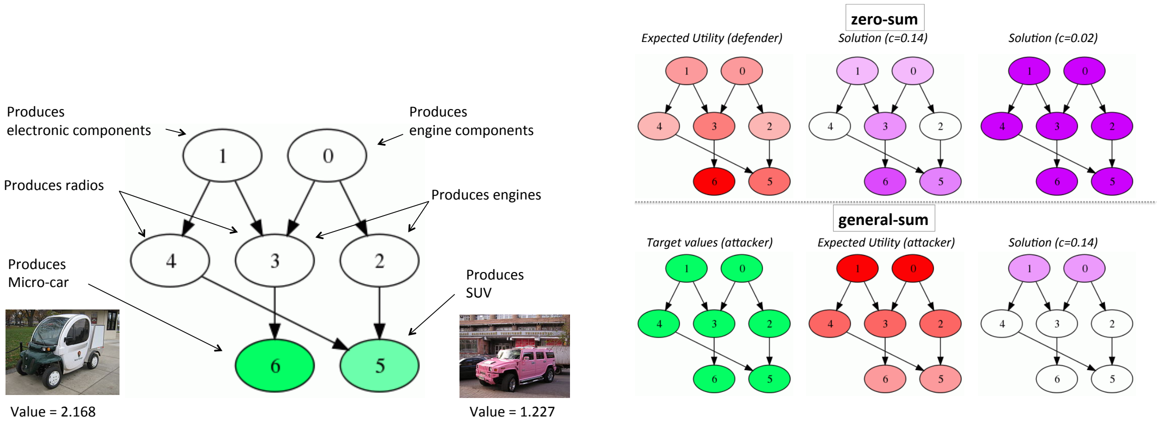


Figure 3.1. A simple supply chain example. Left: supply chain and defender worths for targets (darker means higher values). Right: solutions for the zero-sum (top) and general-sum (bottom) variants.

this case, becomes faulty), $z(o,t)$. We let $z(o,t) = 1$ when node t is not inspected and $z(o,t) = 0$ when it is. Next, we must specify the contagion probabilities for each edge. We use $p_{t,t'} = 0.5$ for all edges here, and assume that they are independent of security decisions Moreover, we assume that the attacker only attacks a single target.

The results are color coded in Figure 3.1: the darker colors correspond to more valuable nodes. Note that while intrinsic worth is only ascribed to the final products, all components carry some value, due to their indirect impact on the final product (for example, a faulty part will, with some probability, make the component which uses it faulty as well). First, suppose that the game is zero-sum. We show the results for two different inspection costs, $c_{high} = 0.14$ and $c_{low} = 0.02$ in Figure 3.1 (right, top). The higher cost setting (Figure 3.1, right, top, middle solution) yields a security configuration in which five of the seven nodes incur some probability of inspection, with the heavier colors corresponding to a higher inspection probability. The low-cost setting (Figure 3.1, right, top, solution on the right) yields a solution in which every node is defended with probability 1. Next, consider a non-zero-sum variant in which the defender's utility is as before, while the attacker has uniform valuations (worths) over targets. The solution for this case with cost 0.14 is shown in Figure 3.1, right, bottom (the figure also shows the attacker's worths, as well as expected utilities derived from the dependency graph). This solution would at first sight seem quite unintuitive: the defender defends *only* the two targets at the top, which have the least value to him! The reason is that these targets happen to have the highest expected utility for the attacker, since they result in the greatest utility from cascades, because the attacker's worths are identical for all targets. The defender will partially defend these targets, and given the defender's strategy, the attacker will still prefer to attack one of these, but will now be caught with positive probability.

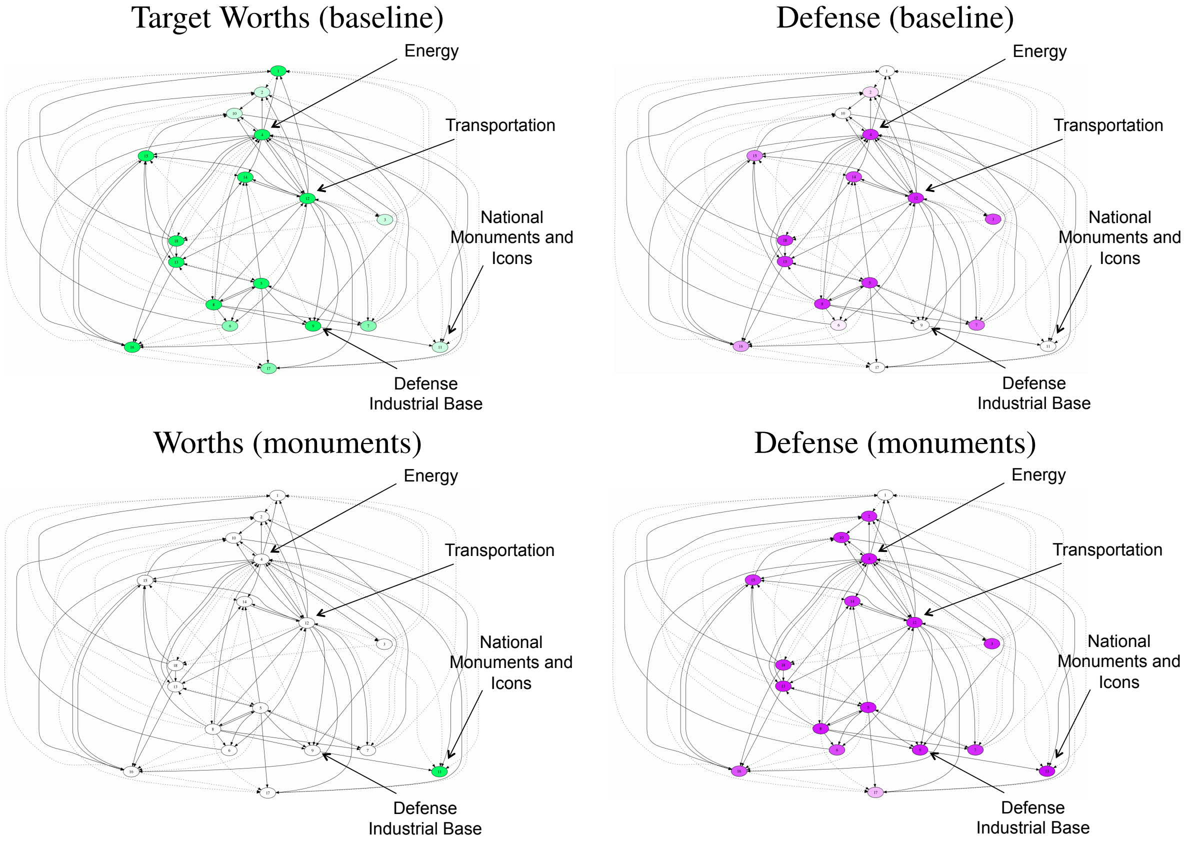


Figure 3.2. Defending critical infrastructure and key resources. Top: baseline, with node worths based on rough economic impact. Bottom: an anomalous valuation function where only monuments and icons sector has positive worth.

3.5.2 Defending Critical Infrastructure: The Lobby Effect

Our second illustration of the framework developed above is on a graph representing dependencies between the critical infrastructure and key resource sectors listed on the DHS and FEMA websites. We used these websites to also infer the dependencies between the sectors, as well as the relative strengths of these dependencies. We then grouped these into "high" and "low" strength, with cascade probability set to 0.5 in the former and 0.1 in the latter cases. Defense cost is fixed at $c = 0.2$, and when a target is defended, it is assumed that no direct attack on it can succeed, while an attack on an undefended target succeeds with probability 1.

Figure 3.2 offers a view of the defense configuration in two cases: first (top), the baseline case in which importance of nodes is roughly representative of its economic value, and second (bottom), a comparative example in which only the monuments and icons sector is deemed valuable. One motivation for this particular contrast is to illustrate a lobby effect which makes the value of a particular sector appear "out-of-whack" with economic considerations.

One interesting observation is that in the baseline case, even though every node has positive worth, not all nodes are defended with positive probability. For example, the defense industrial base sector is left undefended, as is the monuments and icons sector. In contrast, if there is a highly effective lobby on behalf of monuments and icons, to one's surprise *nearly all nodes are fully defended*, and defense expenditures are *much higher than in the baseline case*. This difference is due to the nature of dependencies: monuments and icons has either direct, or indirect but strong dependencies on almost all other sectors. The broader policy insight we may glean is that lobbying can have compounding effects on the budget, and a global impact well beyond what is intended by the direct lobbying effort due to systemic interdependencies.

3.6 Experiments

The goal of this section is to illustrate the value of our framework as a computational tool for designing security in interdependent settings. Specifically, we aim to demonstrate that our approach clearly improves on state-of-the-art alternatives, and offers a scalable solution for realistic security problems. We pursue this aim by randomly constructing dependency graphs using Erdos-Renyi (ER) and Preferential Attachment (PA) generative models [84], as well as using a graph representing a snapshot of Autonomous System (AS) interconnections generated using Oregon routeviews [87]; this graph contains 6474 targets and 13233 edges and thus offers a reasonable test of scalability. In the ER model, every directed link is made with a specified and fixed probability p; we refer to it as ER(p). The PA model adds nodes in a fixed sequence, starting from an arbitrary seed graph with at least two vertices. Each node i is attached to m others stochastically (unless $i \leq m$, in which case it is connected to all preceding nodes), with probability of connecting to a node j proportional to the degree of j, d_j.

For the randomly generated networks, all data presented is averaged over 80-100 graph samples. Since we generate graphs that may include undirected cycles, we obtain expected utilities for all nodes on a given graph using 1000-10,000 simulated cascades (below we show that this is more than sufficient). Intrinsic worths w_t are generated uniformly randomly on $[0,1]$. Cascade probabilities $p_{t,t'}$ (when independent of security strategies) were set to 0.5 unless otherwise specified. Except where otherwise specified, we restrict the defender to two security configurations at every target, one with a cost of 0 which stops attacks with probability 0 and one with a cost of c which prevents attacks with probability 1.

Where relevant, we run local search starting from 20 random starting points in addition to the three described above, unless specified otherwise. Finally, unless otherwise specified, we consider games with 50 targets for the general setting, and 100 targets for the restricted setting with security-independent cascades. We note that even with only 50 targets the running time of local search with random restarts on a given game instance was on the order of hours for large L. A single data point in many experiments below is therefore a product of as much as 400 processor-hours.

3.6.1 Sampling Efficiency

Throughout our experiments we use 10,000 samples to evaluate the expected utilities of players. A natural question is: are we taking enough samples? To answer this, we systematically varied the number of samples between 0 (i.e., letting $U_{o,t} = -w_t$) and 100,000. Our results offer strong evidence that 10,000 samples is more than enough: the expected utility (evaluated using 100,000 samples) of the resulting defense configurations becomes flat already when the number of samples is 1000.

3.6.2 Scalability

An important question given the complexity of our framework is whether it can scale to realistic defense scenarios. To test this, we ran our *restricted* framework (i.e., a single target of attack and security-independent cascades) on the AS graph consisting of 6474 targets and 13233 edges. Since this is a large undirected graph containing cycles, a sampling approach was required, but the total running time (including both sampling and solving linear programs) amounted to less than 1 hour. Given the importance of security, and the fact that *distributions* of security settings are computed once (or at least infrequently, as long as significant changes to the interdependency structure are not very frequent), this seems a relatively small computational burden.

3.6.3 Comparison to State-of-the-Art Alternatives

There are two prime computational alternatives to our framework. The first is to assume that targets are independent. While it is not difficult to show that in the worst case this can be quite a poor approximation, we offer empirical support to the added value of our approach below. The second is to use a well-known heuristic developed in the context of vaccination strategies on networks. This latter heuristic would in our case defend nodes in order of their connectivity (degree), until the defense budget is exhausted. Figure 3.3 (left) compares our approach in the restricted setting (single-target attack and security-independent cascades) to the former, while Figure 3.3 (middle, right) compares it to the latter. In both cases, computing optimal defense strategies using our framework yields much higher utility to the defender than the alternatives.

In the general case, one trivial way to compute an optimal solution is to search all possible defender (leader) actions, compute the best response of an attacker, and choose the action for the defender maximizing his utility. This trivial approach is linear in the size of the game. The problem is that the game size grows exponentially with the number of targets. Here we compare our simple local search routine with no random restarts to the optimal search in terms of running time and expected attained utility for the defender. The comparison is done in a simplified setting where we generate networks of interdependencies according to an Erdos-Renyi generative model with edge probability 0.4. We fix cascade probabilities to be $p_{t,t'} = 0.2$ whenever there is an edge between t and t' and t' is not defended; when t' is defended, we set $p_{t,t'} = 0$. We also fix defense costs at $c = 0.2$ and limit attacks to a single target ($L = 1$). Figure 3.4 (left) shows that local search is

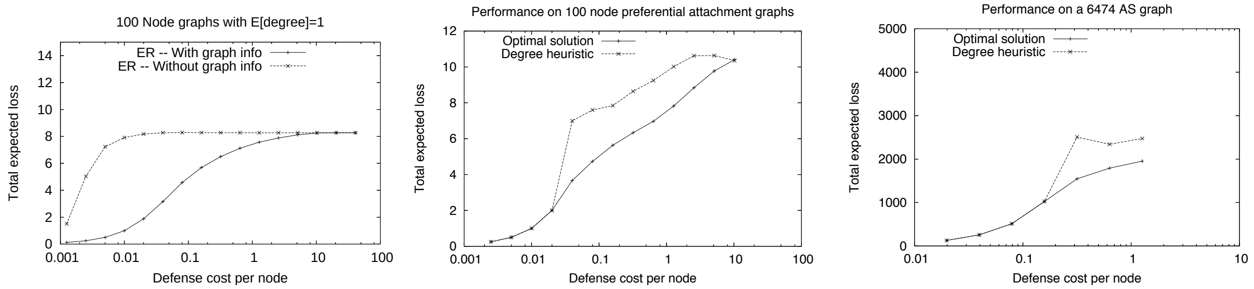


Figure 3.3. Left: Comparison between our approach ("with graph info") and one assuming independence ("without graph info") using the ER(0.1) generative model. Middle/right: Comparison of total expected loss (disutility to the defender) with the degree-based heuristic in the restricted setting. Left: On PA graphs. Right: On the AS graph.

dramatically more scalable; indeed, optimal search quickly becomes intractable. Figure 3.4 (right) demonstrates that there are no (statistically significant) differences between the optimal objective value and that of the local search solution (confidence intervals omitted for clarity).

Since our model of security is partly motivated by epidemic spread (e.g., bioterrorism), it is natural to compare our approach to targeted vaccination on networks (widely recognized as state-of-the-art when initial infections are random [93, 80, 53]), where nodes are defended in decreasing order of degree.[4] Figure 3.4 (right) shows that the *targeted vaccination* heuristic performs significantly worse than local search, even when we completely remove inoculated nodes from the network.

Aside from interdependencies, two other important aspects of our model are the fact that it allows an arbitrary number of security configurations, instead of simply allowing the defender to defend, or not, each target, and its ability to optimize with respect to both intelligent attackers and inadvertent failures. We now show that both of these can add substantial value. Figure 3.5 (left) shows a comparison between a solution which only allows two configurations (defend and do not defend) and two solutions which also allow for a third configuration, which is less effective than full defense, but also less costly. We consider two potential third options, one providing 50% defense at 12.5% of the cost of full defense (1/2 – 1/8) and one providing 75% defense at 12.5% cost (3/4 – 1/8). It is clear from this graph that considering the third configuration adds considerable value. Figure 3.5 (right) assumes that all (or nearly all) failures arise randomly, and compares a solution which posits an attacker to an optimal solution. Again, the value of solving the problem optimally is clear. This plot actually shows an interesting pattern, as the expected utility of the defender is non-monotonic in cost when the solution is suboptimal. This is because the differences between the two solutions are most important when costs are intermediate; with low costs, nearly everything is fully defended, while high costs imply almost no defense.

[4] There are a plethora of minor variations on this general heuristic, but the performance of the best tends to be similar to this baseline.

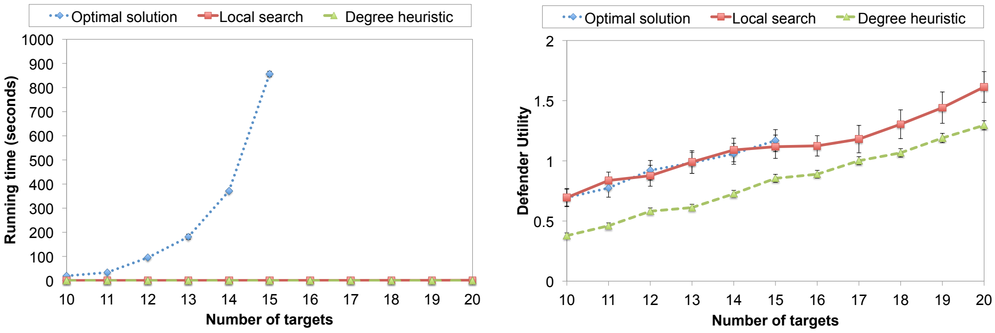


Figure 3.4. Comparison between local search, optimal search, and targeted (degree-based) vaccination. Values are generated according to a Pareto distribution with $\gamma = 1.1$ (the results are robust to variations of this distribution and other parameters). Left: runtime comparison. Right: utility comparison.

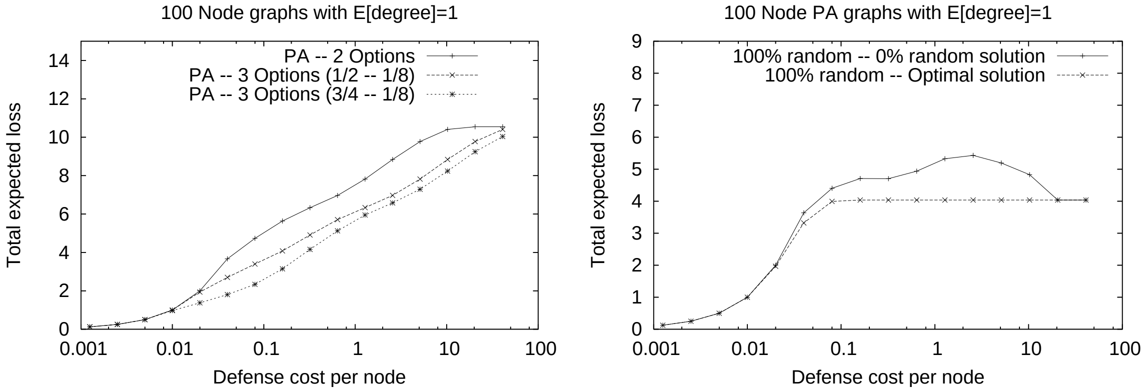


Figure 3.5. Left: Comparison between assuming only two configurations, and allowing the defender to consider three alternatives. Right: Comparison between a solution which assumes that failures are only due to attacks, and an optimal solution, when failures are actually random. Comparisons use PA graphs.

3.7 Applications to Interdependent Security Analysis

In this section we apply our framework to several network security domains. For simplicity, we restrict attention to zero-sum security games. As above, we consider ER and PA generative models, although we utilize a generalized version of PA. In a generalized PA model, connection probabilities are $\frac{(d_i)^\mu}{\sum_j (d_j)^\mu}$, such that when $\mu = 0$ the degree distribution is relatively homogeneous, just as in ER, $\mu = 1$ recovers the "standard" PA model, and large values of μ correspond to highly inhomogeneous degree distributions. Throughout, we use $\mu = 1$ unless otherwise specified. All parameters are set as in the experiments section, unless otherwise specified. In addition to the generative models of networks, we explore two networks derived from real security settings: one with 18 nodes that models dependencies among critical infrastructure and key resource sectors (CIKR), as inferred from the DHS and FEMA websites, and the second with 66 nodes that captures payments between banks in the core of the Fedwire network [102]. For the CIKR network, each node was assigned a low, medium, or high worth of 0.2, 0.5, or 1, respectively, based on perceived importance (for example, the energy sector was assigned a high worth, while the national monuments and icons sector a low worth). Each edge was categorized based on the importance of the dependency (gleaned from the DHS and FEMA websites) as "highly" or "moderately" significant, with cascade probabilities of 0.5 or 0.1 respectively. For the Fedwire network, all nodes were assigned an equal worth of 0.5, and cascade probabilities were discretely chosen between 0.05 and 0.5 in 0.05 increments depending on the weight of the corresponding edges in [102].

3.7.1 The Impact of Uncertainty

Our framework offers a natural way to incorporate uncertainty about the network into the analysis. An important question is: how much impact on defender decision does uncertainty about the network have? Figure 3.6 quantifies the impact of uncertainty on the quality of defense if the observed graph is the PA network with average degree of 2. When cascade probabilities are relatively high ($p_{t,t'} = 0.5$ for all edges, top plot), even if the amount of noise is relatively small ($\varepsilon = 0.01$), the resulting increase in the number of possible cascade paths in the network makes the defender much more vulnerable. With smaller cascade probabilities ($p_{t,t'} = 0.1$, bottom plot), however, noise has relatively little impact. It can thus be vital for the defender to obtain an accurate portrait of the true network over which failures may cascade when the interdependencies among the components are strong.

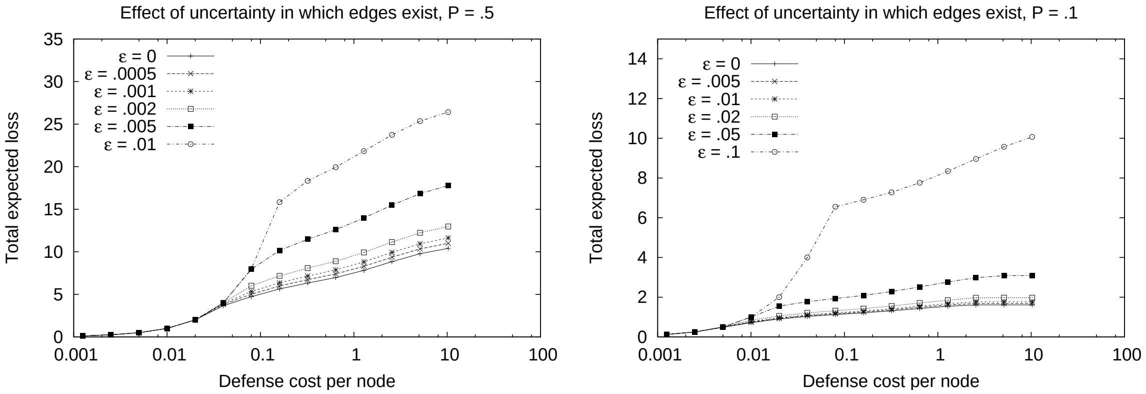


Figure 3.6. The impact of noise on PA networks. Top: when $p_{t,t'} = 0.5$; Bottom: when $p_{t,t'} = 0.1$.

3.7.2 The Impact of Marginal Defense Cost

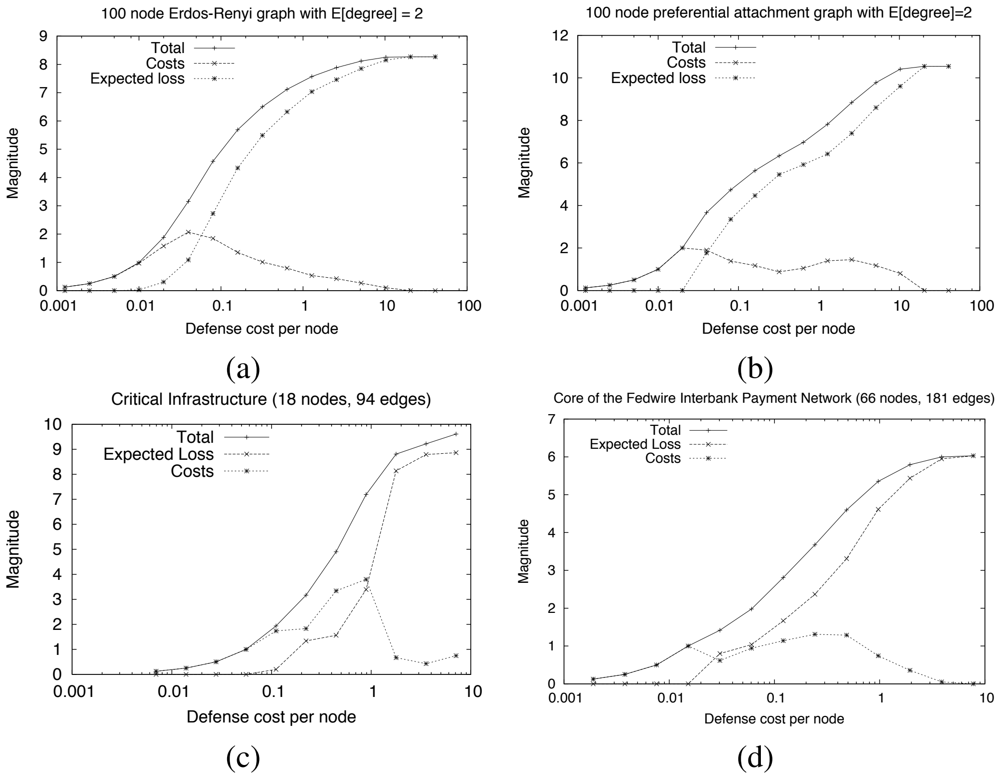
Our first analysis deals with the impact of marginal defense cost c on total defense expenditures (*total costs*), total losses due to failure cascades (or simply *total loss*), and total expenses incurred (or simply *total expense*, corresponding to negative defender utility, or the sum of total costs and total loss). The results for ER and BA (both with 100 nodes and average degree of 2), as well as CIKR and Fedwire networks are shown in Figure 3.7. All the plots feature a clear pattern: expected loss and (negative) utility are monotonically increasing, as expected, while total costs start at zero, initially rise, and ultimately fall (back to zero in 3 of the 4 cases). It may at first be surprising that total costs eventually fall even as marginal costs continue to increase, but this clearly must be the case: when c is high enough, the defender will not wish to invest in security at all, and total costs will be zero. What is much more surprising is the presence of two peaks in PA and Fedwire networks. Both of these networks share the property that there is a non-negligible fraction of nodes with very high connectivity [84, 102]. When the initial peak is reached, the network is fully defended, and as marginal costs rise further, the defender begins to reduce the defense resources expended on the less important targets. At a certain point, only the most connected targets are protected, and since these are so vital to protect, total costs begin increasing again. After the second peak is reached, c is finally large enough to discourage the defender from fully protecting even the most important targets, and the subsequent fall of total costs is no longer reversed.

3.7.3 Changing the Number of Attacked Targets

Our next analysis concerns an important extension that traditional Stackelberg security game approaches cannot handle in a scalable way: allowing an attacker to attack more than a single target. Specifically, we study the impact of the number of targets L an attacker can attack on total defense expenditures, total losses due to failure cascades, and total expenses incurred. We do this while keeping cascade probabilities $p_{t,t'}$ independent of defense configuration; we set all of these to $p = 0.2$. Moreover, we generate the dependency graphs based on the Erdos-Renyi generative model with edge probabilities fixed at 0.05.

Total defense expenditures (costs) are shown in Figure 3.8 (left) for three different values of cost per target defended, c (we also call this marginal defense cost). The difference between the three cost regimes is negligible when only a single target can be attacked, yet the behavior of defense expenditures as L increases exhibits striking qualitative differences, and techniques that only consider $L = 1$ would therefore be blind to these. In all three cases, there is a critical threshold L_c of the number of attacked targets. When $L < L_c$, defense expenditures remain very low and relatively stable, but when $L \approx L_c$, expenditures rise sharply, ultimately leveling off at a much higher value which again remains relatively stable for $L > L_c$. Surprisingly, increasing marginal defense cost c causes L_c to increase: it takes greater attacker capability to stimulate the defender to invest more in security; however, the rise in security investment is greater for higher c once the threshold L_c is reached.

Figure 3.8 (right) shows the total loss as a function of the attacker's capability L. The result is



(a)

(b)

(c)

(d)

Figure 3.7. Expected loss, cost, and their sum in (a) 100-node ER(0.2), (b) 100-node PA, (c) 18-node critical infrastructure, and (d) 66-node core of the Fedwire networks as defense cost increases. The results for ER and PA are averages over 100 stochastic realizations of these networks.

somewhat counter to initial intuition: the total losses are non-monotonic. The reason comes from the observation we had already made about total expenditures: until a threshold L_c is reached, few defense resources are deployed, and total losses rise, but after the threshold, defense expenditures ramp up substantially, and, as long as c is sufficiently low, the defender will ultimately come to defend every target. The pattern of total defender expenses (the sum of losses and total expenditures; not shown) is largely predictable: expenses increase monotonically with L, and are higher for higher c.

3.7.4 Resilience to Targeted Attacks: The Impact of Network Structure

One of the important streams in the network science literature is the question of relative resilience of different network topologies to failures, random or targeted. One feature of network topology, the distribution of degrees (number of node neighbors) has received particular attention. There is, in particular, one measure of degree distribution—its *homogeneity*—that plays an especially important role. (For example, an Erdos-Renyi network has a homogeneous degree distribution,

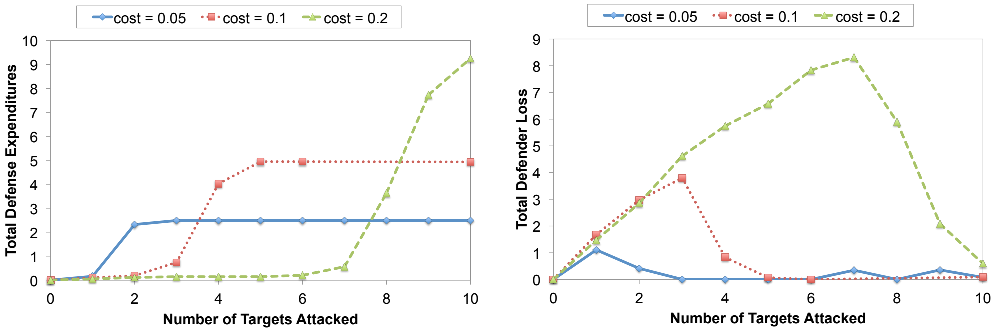


Figure 3.8. Total defense expenditures (left) and losses due to cascading failures (right) as the number of attacked targets increases for three different defense cost values (i.e., cost of defending a single target): 0.05, 0.1, and 0.2.

while a heavy-tailed distribution, such as Pareto, is inhomogeneous.) Two very disparate streams of literature tie homogeneity of the degree distribution to network resilience. The first of these features a widely replicated finding that networks with an inhomogeneous (e.g., scale-free) degree distribution exhibit poor tolerance to targeted attacks as compared to Erdos-Renyi graphs [6, 84]. On the other hand, when failures are random (no attacks), scale-free graphs have been found to be more resilient than Erdos-Renyi counterparts. The second stream of literature demonstrates that scale-free graphs are particularly easy to defend against epidemic spread, as inoculating high-degree nodes dramatically reduces the expected number of infections; however, this stream does not model targeted attacks.

Our framework allows us to cleanly unify both these streams of literature and present a much more refined analysis of the relationship between the homogeneity of the degree distribution and network resilience to cascading failures. Specifically, we undertake here a study of the total losses and costs incurred by the defender under a variety of network regimes.

As a starting point, consider Figure 3.9 (left), which shows the defender's utility for three different network topologies, PA, ER, and Fedwire as a function of cost c. The results presented in this figure are generated based on our special case when cascade probabilities $p_{t,t'}$ are independent of security decisions, and when the attacker can only attack a single target (through the rest of this paper, we focus only on the impact of deliberate attacks and fix the probability of "nature" to 0). In light of the previous discussion, what we can readily observe in Figure 3.9 (left) would appear quite remarkable: network topology seems to play little role in resilience. A superficial difference here is that we consider a cascading failure model, while most of the previous work on the subject involving targeted attacks focused on diminished connectivity due to attacks. We contend that the most important distinction, however, is that previous work studying resilience did not account for a simple observation that most important targets of potential attacks are also most heavily defended; indeed, to the best of our knowledge, none of the previous work on resilience in the face of attacks allows for endogenous defense decisions. Indeed, we can observe from the figure that once defense costs c are sufficiently high, PA leads to substantially higher losses (greater disutility to the defender), confirming previous results in this rather extreme setting.

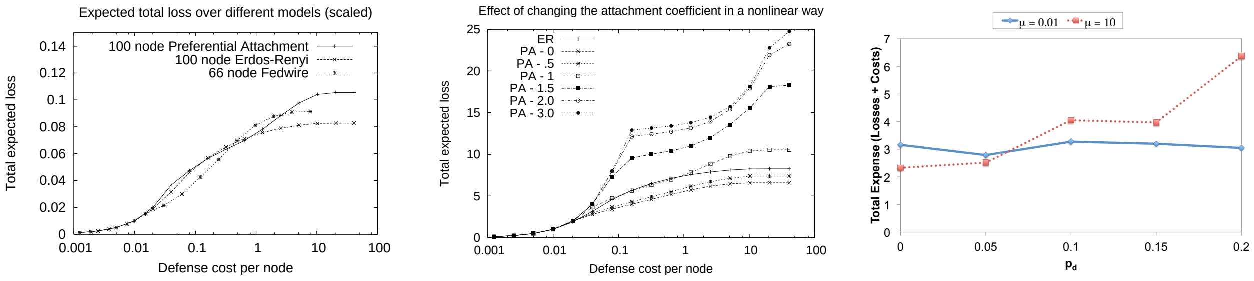


Figure 3.9. Left: Expected total loss: comparison across different network structures. Middle: Expected defender disutility in the generalized PA model as we vary μ (keeping average degree fixed at 2). ER is also shown for comparison. Right: Total defender expenses (total expenditures + losses from cascades) as a function of p_d for $\mu = 0.01$ (nearly Erdos-Renyi) and $\mu = 10$ (highly hub-like structure). Cascade probabilities of undefended nodes are fixed at $p = 0.2$. Cost of defending each node is fixed at 0.5.

To investigate the impact of network topology on resilience further, we consider the generalized PA model in which we systematically vary the homogeneity of the degree distribution by way of the parameter μ. Figure 3.9 (middle) shows the results for the special case of security-independent cascades with the attacker restricted to attack only one target. In this graph, we do observe clear variation in resilience as a function of network topology, but the operational factor in this variation is *homogeneity in the distribution of expected utilities, rather than degrees*: increasing homogeneity of the utility distribution *lowers* network resilience. This seems precisely the opposite of the standard results in network resilience, but the two are in fact closely related, as we now demonstrate. Superficially, the trend in the figure seems to follow the common intuition in the resilience literature: as the degree distribution becomes more inhomogeneous (more star-like), it becomes more difficult to defend. Observe, however, that ER is actually more difficult to defend than PA with $\mu = 0$. The lone difference of the latter from ER is the fact that nodes that enter earlier are more connected and, therefore, the degree distribution in the PA variant should actually be more *inhomogeneous* than ER! The answer is that random connectivity combined with inhomogeneity of degrees actually makes the distribution of *utilities* less homogeneous in PA with $\mu = 0$, and, as a result, fewer nodes on which defense can focus as compared to ER. On the other hand, as the graph becomes more star-like, the utilities of all nodes become quite similar; in the limiting case, all nodes are only two hops apart, and attacking any one of them yields a loss of many as a result of cascades.

Our final exploration in this vein considers a more general setting where security decisions have some (varying) effect on the likelihood of cascade spread. Specifically, define the parameter p_d as the probability that a cascade spreads to a node which is defended, and fix the probability that a cascade spreads to an undefended node at 0.2. Thus, if $p_d = 0$, we have an instance of perfect inoculation: if a node is defended (inoculated), it is equivalent to removing that node from the network entirely. At the other end of the spectrum, $p_d = 0.2$ will imply that defense has no impact on the probability of cascades. Figure 3.9 (right) presents the total defender disutility (losses due to cascading failures + defense costs incurred) as a function of p_d for two extreme cases of μ, one ($\mu = 0.01$) corresponding to a highly homogeneous degree distribution, while the other ($\mu = 10$) to a highly inhomogeneous one. The two classes of graphs exhibit dramatically different resilience behavior as a function of p_d which paints a more complete picture than the literature on network resilience to date. When $p_d = 0.2$ (equal to the cascade probability when a node is not defended), hub-like structures are far less resilient to targeted attacks as compared to a graph with a homogeneous degree distribution; this is inline with previous results, which suggest that inhomogeneous graphs are less resilient [6]. With $p_d = 0$, on the other hand, hub-like networks are highly resilient, since it suffices for the defender to target the few hubs; this is similar to the observation that targeted vaccination is more effective on scale-free graphs [93], although in that stream of literature failures are assumed to arise randomly, rather than in a targeted manner. At the high level, the resilience of the hub-like network decreases with increasing p_d, whereas a homogeneous network remains relatively unaffected by p_d. The reason is that when p_d is high, a hub-like structure implies low diameter. Unless the hub itself is actually removed from the network by the defense action, it can serve as the conduit for failure cascades started at other nodes; therefore, when p_d is high the defense of the hub is insufficient to make the network resilient, and vastly greater defense expenditures are required. In contrast, a homogeneous network has no such hubs with global connectivity, and is therefore less sensitive to p_d.

There is another aspect of network topology that has an important impact on resilience: network density. Figure 3.10 (left) shows a plot of an Erdos-Renyi network with the probability of an edge varying between 0.0025 to 0.08 (average degree between .25 and 8) and cost c fixed at 0.04. Clearly, expected utility and loss of the defender are increasing in density, but it is rather surprising to observe how sharply they jump once the average degree exceeds 1 (the ER network threshold for a large connected component); in any case, network density has an unmistakable impact. The reason is intuitive: increased density means more paths between targets, and, consequently, greater likelihood of large cascades in the event that a target is compromised. Total cost initially increases in response to increased density, in part to compensate for the increased vulnerability to attacks, but eventually falls, since it is too expensive to protect everything, and anything short of that is largely ineffective.

3.7.5 Interaction Between Cascade Probabilities and the Number of Targets Attacked

In this section we study the impact of the cascade probability to a defended node, p_d, while at the same time varying the attacker's capability L. As in the previous section, we maintain the probability that a failure cascades to an undefended node at 0.2. We generate the dependency graphs based on the Erdos-Renyi generative model with edge probabilities fixed at 0.05.

Figure 3.10 (right) shows the total defense expenditures (cost per target defended fixed at 0.1).

While the differences are relatively small, there is a clear pattern: when the number of targets attacked is low (below L_c), increasing the impact of defense on cascade probability prompts the defender to increase investment in security (defense has an increasing marginal value), but once attacker capabilities are high, defense expenditures fall when p_d falls (i.e., defense has higher impact). In the latter case, making the network sufficiently resilient to attacks requires relatively fewer protected nodes and, therefore, lower defense expenditures. Indeed, decreasing p_d systematically reduces total defender expenses (sum of losses due to cascades and defense costs).

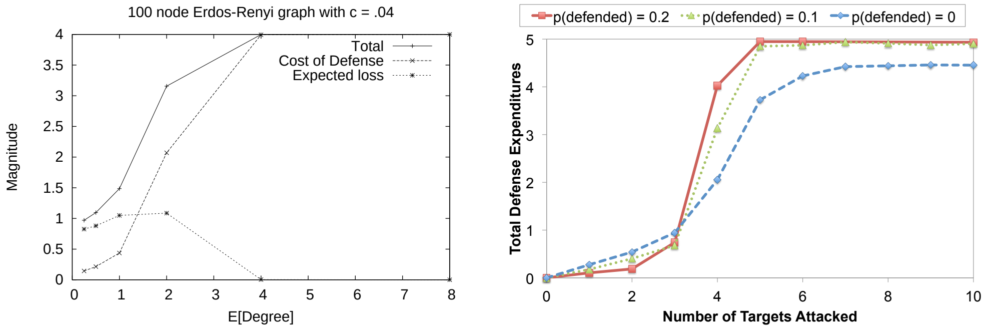


Figure 3.10. Left: Expected loss, cost, and their sum in 100-node Erdos-Renyi networks as a function of network density (equivalently, expected degree). Right: Total defense expenditures as the number of attacked targets increases for three different values of p_d (p(defense) in the legend): 0.2 (cascades independent of defense), 0.1 (defense partially protects from cascades), and 0 (defense fully protects from cascades). Graphs are ER(0.05).

3.8 Conclusion

We presented a framework for computing and approximating optimal security policies in network domains. Our framework involves a general model of asset interdependencies, which we instantiate using a dependency graph between assets and a cascading failures model based on a common epidemiological model of disease contagion. In the general case, we offer an effective approximation technique based on a combination of submodular optimization and a local search heuristic. Moreover, we show that in an important special case which restrict the attacker's capabilities to only attack one target and restricts the cascade probabilities to be independent of security decisions, we can effectively decouple simulations that estimate player expected utilities from a linear programming formulation which subsequently computes an optimal security policy. Our results demonstrate the value of our approach as compared to alternatives, and show that it is scalable to realistic security settings. Furthermore, we used our framework to analyze four models of interdependencies: two based on random graph generation models, a simple model of interdependence between critical infrastructure and key resource sectors, and a model of the Fedwire interbank payment network.

Chapter 4

Stochastic Stackelberg Games, with Applications to Adversarial Patrolling

4.1 Introduction

Game theoretic approaches to security based on Stackelberg game models have received much attention in recent years, with several finding deployment in real-world settings including LAX (Los Angeles International Airport), FAMS (United States Federal Air Marshals Service), TSA (United States Transportation Security Agency), and USCG (United States Coast Guard) [63, 12]. At the backbone of these applications are defender-attacker Stackelberg games in which the defender first commits to a randomized security policy, and the attacker uses surveillance to learn about the policy before attacking. The analysis of Stackelberg security games has focused primarily on computing a Strong Stackelberg equilibrium (SSE) [40, 92, 67].

To date, the Stackelberg game models for all real-world security applications assume that attacker knows the probability that each target is covered by the defender, but is oblivious to the actual sequence of defender moves. For example, the defender may in fact visit targets according to some fixed (but randomly generated) patrolling schedule, but the attacker is presumed to be unable to observe the defender's location at any point during the patrol. In many realistic settings, such as USCG [12], it is likely that the attacker can in fact observe the patrol while it is in progress (e.g., the coast guard ships can be quite overt). Thus, a more plausible model in such a setting would allow the attacker to observe both the randomized policy of the defender (i.e., probability distribution over moves) as well as current defender location.

We formally model this setting as an *adversarial patrolling game*, or APG. An APG is a very special case of a much broader class of *general-sum discounted stochastic Stackelberg games (SSGs)*. An SSG involves a leader, who commits to a (possibly stochastic) policy (in general, a function of all previous actions and states), and a follower, who observes the leader's commitment and optimally responds to it. We begin by studying the properties of Strong Stackelberg Equilibria in SSGs (Section 4.3), and proceed to offer algorithms for computing exact and approximate SSE solutions when the leader is restricted to play Markov stationary policies (Sections 4.4 and 4.4.2).

Subsequently, we proceed to offer an extensive treatment of adversarial patrolling games, mostly restricting attention to their zero-sum variants. We consider three different general models of adversarial patrolling. The first, baseline, model described in Section 4.2.2 is that of the most basic adversarial patrolling scenario, with an exogenously imposed network of constraints on defender's moves, a single defender resource (i.e., a single patroller), and an attacker who can deploy an attack one time step after the actual decision to attack is made. We describe an exact NLP formulation of this problem. We then extend this model, and the baseline NLP formulation, in two directions. First, we generalize the formulation in Section 4.6.3 by allowing the defender to use multiple defense resources. In this case, defender's actions take the form of coverage vectors (i.e., specify which targets are covered). Second, the baseline formulation is generalized in Section 4.6.3 to allow the attacks to take more than a single time step to unfold. In this case, the defender must, in general, condition his policies on sequences of several previously visited targets, rather than just the last. As this implies severe limitations on scalability of the approach, we present a formulation which allows a tunable approximation, allowing one to make the best tradeoff between optimality and scalability. In the experimental section (Section 4.7) we compare our approaches to the state-of-the-art alternative, as well as to each other. We also show that the MILP approximation requires only a very coarsely discretized probabilities to obtain near-optimal solutions, and demonstrate the tradeoffs between scalability and approximation quality using our formulation when attacks take more than a single time step.

Our second general model of patrolling relaxes the assumption that the graph which constrains patrolling moves is exogenous, and instead allows the defender to first build the edges, at some cost, that will impose patrolling constraints, and then use this graph in an optimal patrol (Section 4.8). If the baseline formulation were generalized directly, this new setting would introduce integer variables into the non-linear program. We therefore offer an alternative formulation which allows us to relax the integrality constraints.

Our final model of patrolling is a more significant departure from the baseline APG, as it eliminates the graph as a constraint altogether, imposing instead differential costs on the defender for making specific moves between targets (Section 4.9). Since the defender's objective now involves minimizing patrolling costs, the game is no longer zero-sum. As using integer variables in this APG variant becomes inevitable, we generalize the MILP approximation for this setting.

4.1.1 Related Work

Our work lies at the point of convergence of several research thrusts: pursuit-evasion games, inspection games, robotic patrolling (particularly, in adversarial settings), stochastic games, and Stackelberg security games.

Pursuit-evasion (alternatively, hider-seeker, infiltration, or search) games typically involve a hider, either stationary or mobile, who hides in or traverses a path through a graph, and a seeker, whose goal it is to find the hider [50, 90, 52, 9, 8, 1, 60]. A number of variations on that general theme have been considered, some studying the number of seekers required to find the hider with certainty [90], others aiming to minimize the expected time to find the seeker [52, 9, 1], yet others considering a game where both players simultaneously choose a vertex on a graph, with the distance between the ultimate choices determining the amount one pays to the other [36]. Closely related infiltration games involve a hider who traverses a path and a seeker who chooses a set of vertices that overlap with the hider as much as possible [8]. Another closely related class of games is *accumulation games*, which is a game between a hider, who distributes a divisible resource among a collection of descrete locations (e.g., nodes on a graph), and a seeker, who aims to find a sufficient fraction of the resource by searching a limited number of locations [68, 10]. The goal of most of this work is to bound the value of the game (as a function of the graph) and, if possible, to mathematically characterize player strategies. In contrast, [58] present exact and an approximate double-oracle algorithms for *computing* equilibrium hider-seeker strategies. There are several salient differences between our setting and approach and the literature on search games. First, we consider general stochastic games, whereas search games can be viewed (modulo a few technical nuances) as special cases. Second, the attacker the special case of APGs that we study can choose whether to attack, and when, and is able to condition his choice on observed location of the defender. Third, we focus on leader-following games, allowing the attacker full knowledge of the defender's policy, and in several instances allow these to be general-sum. A similar line of work has had a long history in the Operations Research community, commony referred to as *network interdiction* [111, 41, 112, 24, 25, 82]. Like our work, these are focused on mathematical programming formulations to compute optimal policies, but, unlike us, the focus is on damaging the network to reduce network flow (or increase shortest path), rather than patrolling a network of targets.

Another related thrust is the literature on inspection games [16]. The most basic variant of an inspection game involves an inspectee (e.g., a tax evader) who can choose to perform an illegal or a legal action, and an inspector, who receives a noisy signal upon which he can inspect (at some cost), or not. One qualitative difference between this generic inspector game and our setting is that in our case the defender (inspector) acts first, and the attacker (inspectee) acts *after observing the defender's decision* (which may be randomized, in which case the attacker observes the probability distribution). Moreover, the inspection games feature very simple defender and attacker strategy spaces, whereas strategic complexity is at the root of the problem we study.

The third thrust upon which we build is robotic patrolling. In this literature, there are two distinct approaches. The first, and earliest, is non-adversarial in nature, and most patrolling work falls into this category. In general, classical patrolling work is focused either on covering all the potential targets (or patrol sectors) in the most efficient way, often using multiple coordinated patrollers [2], or on patrolling that minimizes target idleness, or maximizes patrol frequency [7, 35, 55, 48]. The second approach does involve explicit adversarial modeling and in that sense much more like our own. One line of work on adversarial patrolling settings is done in the context of robotic patrols, but involved a very simple defense decision space (for example, with a set of robots moving around a perimeter, and a single parameter governing the probability that they move forward or back) [4, 5, 3]. In a somewhat different vein, [11] study win-lose patrolling games in which the patroller chooses a sequence of targets to visit, while the attacker can only choose a target, and a time of attack, with the goal of mathematically characterizing the value of the game for different classes of graphs. Another line of work on adversarial patrolling studies general-sum patrolling games in which the patroller (defender, leader) first commits to a stochastic policy, which is observed by the attacker who chooses which target to attack and in which context [19, 22, 21, 18, 23, 20]. An important differentiating assumption of the latter work is that, like in our setting, the attacker is assumed to observe both the defender's policy, as well as its past realizations, and can condition his decision on both. Unlike our work, however, Basilico et al., and others, study games in which the attacker is infinitely patient (we consider discounted games in which an attacker can be arbitrarily patient or impatient), and make restrictions on the attacker policy space which we relax. We also study for the first time a number of important variants of the adversarial patrolling problem, one allowing the defender to alter the patrol network at some cost, and another in which a defender incurs variable costs for traversing network edges.

[19], as well as other work that follows in the same framework, build on the literature that explores the problem of computing Stackelberg equilibria in security settings [40, 92, 67, 63, 12]. While most applications of these approaches are to security patrolling problems, they all assume that the adversary cannot observe past realizations of patrol moves. This assumption is quite reasonable in some settings, such as Federal Air Marshall Service, where marshalls are usually not clearly identifiable, but is a strong assumption in others, such as coast guard patrols, which are clearly visible. Our model therefore explicitly allows an attacker to observe past patrol moves.

Finally, in the abstract, Stackelberg models of security are a special case of leader-follower, or Stackelberg games. [40] offered the first extensive treatment of the subject in the setting where games are represented in normal form. [76] present the first results about pure and mixed strategy commitment in the context of finite horizon extensive form games, and [77] study commitment in general stochastic games. Both [76] and [77] offer mainly negative results about mixed-strategy (randomized) commitment, and the latter present an efficient approximate algorithm for computing correlated commitment. As such, ours is the first attempt to compute uncorrelated randomized stationary Markov policies in general discounted stochastic Stackelberg games. Our use of mathematical programming techniques for computing equilibria builds in part on [49], who study the question of computing Nash equilibria in two-player stochastic games. While Stackelberg and Nash equilibria coincide in zero-sum games, they can be very different in general, and we therefore require very different formulations for computing Stackelberg equilibria in stochastic games.

4.2 Stochastic Stackelberg Games

4.2.1 General Setup

We consider two-player infinite-horizon discounted stochastic Stackelberg games (SSGs from now on) in which one player is a "leader" and the other a "follower". The leader commits to a policy that becomes known to the follower who plays a best-response policy. These games have a finite state space S, finite action spaces A_L for the leader and A_F for the follower, payoff functions $R_L(s, a_l, a_f)$ and $R_F(s, a_l, a_f)$ for leader and follower respectively, and a transition function $T_{ss'}^{a_l a_f}$, where $s, s' \in S$, $a_l \in A_L$ and $a_f \in A_F$. The discount factors are $\gamma_L, \gamma_F < 1$ for the leader and follower, respectively. Finally, $\beta(s)$ is the probability that the initial state is s.

The history of play at time t is $h(t) = \{s(1)a_l(1)a_f(1)\dots s(t-1)a_l(t-1)a_f(t-1)s(t)\}$ where the parenthesized indices denote time. Let Π (Φ) be the set of unconstrained, i.e., nonstationary and non-Markov, policies for the leader (follower), i.e., mappings from histories to distributions over actions. Similarly, let Π_{MS} (Φ_{MS}) be the set of Markov stationary policies for the leader (follower); these map the last state $s(t)$ to distributions over actions. Finally, for the follower we will also need the set of deterministic Markov stationary policies, denoted Φ_{dMS}.

Let U_L and U_F denote the utility functions for leader and follower respectively. For arbitrary policies $\pi \in \Pi$ and $\phi \in \Phi$,

$$U_L(s,\pi,\phi) = \mathbb{E}\Big[\sum_{t=1}^{\infty} \gamma_L^{t-1} R_L(s(t),\pi(h(t)),\phi(h(t)))|s(1)=s\Big],$$

where the expectation is over the stochastic evolution of the states, and where (abusing notation)

$$R_L(s(t),\pi(h(t)),\phi(h(t))) = \sum_{a_l \in A_L} \sum_{a_f \in A_F} \pi(a_l|h(t))\phi(a_f|h(t))R_L(s(t),a_l,a_f),$$

and $\pi(a_l|h(t))$ is the probability of leader-action a_l in history $h(t)$ under policy π, and $\phi(a_f|h(t))$ is the probability of follower-action a_f in history $h(t)$ under policy ϕ. The utility of the follower, $U_F(s,\pi,\phi)$, is defined analogously.

For any leader policy $\pi \in \Pi$, the follower plays the best-response policy defined as follows:

$$\phi_\pi^{BR} \overset{\text{def}}{\in} \arg\max_{\phi \in \Phi} \sum_s \beta(s) U_F(s,\pi,\phi).$$

The leader's optimal policy is then

$$\pi^* \overset{\text{def}}{\in} \arg\max_{\pi \in \Pi} \sum_s \beta(s) U_L(s,\pi,\phi_\pi^{BR})$$

Together $(\pi^*, \phi_{\pi^*}^{BR})$ constitute a Stackelberg equilibrium (SE). If, additionally, the follower breaks ties in the leader's favor, these are a Strong Stackelberg equilibrium (SSE).

4.2.2 Adversarial Patrolling Games

Adversarial patrolling games (APGs) form a highly restricted special case of SSGs. We begin with a somewhat restricted definition of these games for clarity, and extend these later. Formally, an adversarial patrolling game can be described by the tuple $\{T, U_d^c(i), U_d^u(i), U_a^c(i), U_a^u(i), \gamma_d, \gamma_d, G\}$, where T is the set of n targets patrolled by the defender, $U_d^c(i)$ and $U_d^u(i)$ are the utilities to the defender if an attacker chooses a target $i \in T$ when it is patrolled and not, respectively, while $U_a^c(i)$ and $U_a^u(i)$ are the corresponding attacker utilities, $\gamma_a, \gamma_d \in (0,1)$ is the discount factor (in some cases, we also allow $\gamma_a = \gamma_d = 1$), and $G = (T,E)$ is a graph with targets as vertices and E the set of directed edges constraining defender patrolling moves between targets. It is useful to consider the representation of this graph as an adjacency matrix A, where $A_{ij} = 1$ if and only if there is an edge from target i to target j. In the special case of zero-sum game APGs which we consider in some detail below, $U_d^c(i) = -U_a^c(i)$ and $U_d^u(i) = -U_a^u(i)$.

The game proceeds in a (possibly infinite) sequence of steps in which the defender moves between targets (subject to the constraints imposed by G), while the attacker chooses the time and target of attack. The defender's (stochastic) patrolling policy is a schedule π which can in general be an arbitrary function from all observed history (i.e., the sequence of targets patrolled in the past) to a probability distribution over the targets patrolled in the next iteration. The attacker is presumed to know the defender's policy π at the time of decision. At each time step t the attacker observes the defender's current location i and may choose to wait or to attack an arbitrary target $j \in T$. If an attacker waits, he receives no immediate utility, while attacking a target j gains the attacker $U_a^c(i)$ if it is covered by the defender at time $t+1$ and $U_a^u(i)$ if it is not. We denote the attacker's policy by a.

We use v_i to denote the expected discounted value to the attacker upon observing the defender at target i. Where relevant, we assume that the defender always starts at target 0, and the aim of the defender is, consequently, to minimize v_0, which the attacker attempts to maximize.

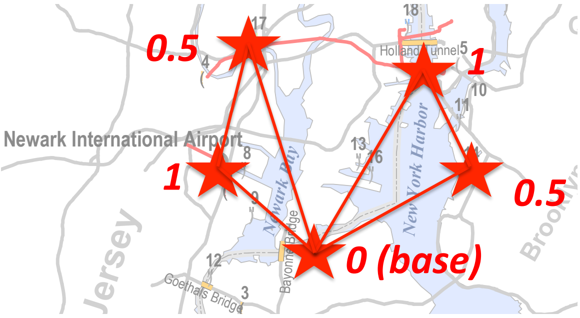


Figure 4.1. Example of a simple New York Bay patrolling scenario.

Example 4.2.1. **USCG's Patrolling Problem as an APG**: USCG safeguards important infrastructure at US coasts, ports, and inland waterway. For this example, we chose a simple Newark Bay and New York Harbor patrolling scenario, shown in Figure 4.1. We chose five possible targets, with the graph roughly representing geographic patrolling constraints (assuming a boat patrol). There is a target which is connected to all others, and is a natural candidate for a base of operations. The number near each target represents its value to the defender and attacker. Two targets have the highest value, but the patrol boat cannot move directly between these. The base has no value intrinsically, but its high connectivity makes it valuable nonetheless (as we shall see below, it will therefore play a crucial role in defense). □

4.2.3 APG as a Stochastic Stackelberg Game

We now show how to formulate an instance of an adversarial patrolling games as a SSG. In our setting, states correspond to the set of targets T (representing the current defender location in the patrol), together with an absorbing state s. Defender actions in each state are the targets j that he can move to in a single time step, while attacker actions are to wait or to attack (for the moment, we will assume that we can compute expected utilities when attacker chooses to attack; we deal with the issue of which targets are attacked below). The state transitions are actually deterministic, conditional on player actions: if the attacker chooses to attack, the system always transitions to the absorbing state s; otherwise, the next target is completely determined by the defender's action. Finally, if the attacker waits, our baseline model involves zero reward accruing to both players. Let R^a_i denote the expected utility to attacker of attacking in state i; the defender's corresponding utility is R^d_i, which becomes $-R^a_i$ if the game is zero-sum. The stochastic game has an infinite horizon, and $\gamma_L = \gamma_d$ while $\gamma_F = \gamma_a$. Figure 4.2 offers a schematic illustration of a zeros-sum APG as a stochastic game.

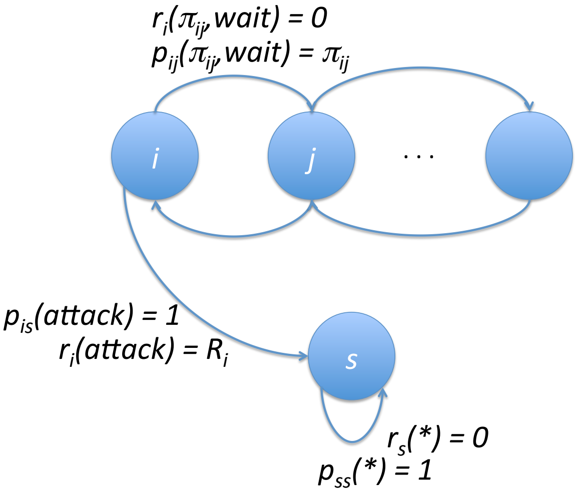


Figure 4.2. Schematic illustration of APG as a stochastic game, showing example targets-states i and j, as well the absorbing state s. $p_{ij}(\cdot)$ denotes the transition probability, as a function of the probability π_{ij} that the defender moves from i to j and whether or not the attacker chooses "wait" or "attack".

4.3 The form of a SSE in Stochastic Games

It is well known that in general-sum stochastic games there always exists a Nash equilibrium (NE) in Markov stationary policies [49]. The import of this result is that it allows one to focus NE computation on this very restricted space of strategies.

To begin, let us state a very basic, and weak, result that does hold in general:

Lemma 4.3.1. *For any general-sum discounted stochastic Stackelberg game, if the leader follows a Markov stationary policy, then there exists a deterministic Markov stationary policy that is a best response for the follower.*

This follows from the fact that if the leader plays a Markov stationary policy, the follower faces a finite MDP. A slightly weaker result is, in fact, at the core of proving the existence of Markov stationary NE: it allows one to define a best response correspondence in the space of (stochastic) Markov stationary policies of each player, and an application of Kakutani's fixed point theorem completes the proof. The difficulty that arises in SSGs is that, in general, the leader's policy *need not be a best response to the follower's*.

We now show that in general an optimal leader policy need not be stationary Markovian and, indeed, a stationary Markovian policy could be arbitrarily suboptimal.

Example 4.3.1. **The leader's optimal policy may not be Markov stationary even if transition probabilities are deterministic and independent of player actions. Moreover, the best stationary policy can be arbitrarily suboptimal.**

Consider the following counterexample.[1] Suppose that the SSG has three states, i.e., $S = \{1,2,3\}$, and the leader and the follower have two actions each, $A_L = \{U,D\}$ for the leader and $A_F = \{L,R\}$ for the leader. Let initial state be $s = 1$ and suppose that the following transitions happen deterministically and independently of either player's decisions: $T_{12} = 1$, $T_{23} = 1$, $T_{33} = 1$, that is, the process starts at state 1, then moves to state 2, then, finally, to state 3, which is an absorbing state. In state $s = 1$ only the follower's actions have an effect on payoffs, which is as follows: $R_L(1,\cdot,L) = -M$, $R_L(1,\cdot,R) = 0$, $R_F(1,\cdot,L) = \varepsilon$, $R_F(1,\cdot,R) = 0$, where M is an arbitrarily large number and $\varepsilon << M$. In state $s = 2$, in contrast, only the leader's actions have an effect on payoffs: $R_L(2,U,\cdot) = R_L(1,D,\cdot) = 0$, $R_F(1,U,\cdot) = -M$, $R_F(1,D,\cdot) = 0$. Suppose that the discount factors $\gamma = \delta$ are close to 1. First, note that a Markov stationary policy for the leader would be independent of the follower's action in state 1, and, consequently, the follower's best response is to play L, giving the leader a payoff of $-M$. On the other hand, if the leader plays U when the follower plays L and D otherwise, the follower's optimal policy is to play R, and the leader receives a payoff of 0. Since M is arbitrarily large, the difference between an optimal and best stationary policy is arbitrarily large.

We can consider an alternative restriction on stochastic games, disallowing the leader to observe actual realizations of the follower's actions (since payoff observations reveal information about joint action choices, we would also make an associated assumption that payoff observations are delayed sufficiently long to be uninformative for decisions; in any case, the result below is negative even if we ignore this complication). However, this restriction does not suffice either.

Example 4.3.2. **The leader's optimal policy may not be Markov stationary even if transition probabilities are deterministic and only depend on the follower's action, and, moreover, the leader cannot condition his policy on past observations of follower's actions (or associated payoffs). Moreover, the best stationary policy can be arbitrarily suboptimal, and the sequence of states on which an optimal policy conditions can be arbitrarily long.**

Consider Example 4.3.1, but modify it as follows: suppose each the follower's action maps to a unique state, so that if follower plays L in state 1, next state is $1a$, while R maps to $1b$. Both $1a$ and $1b$ map deterministically to a sequence of states of arbitrary length to state 2, and the rest of

[1] We are grateful for Vincent Conitzer for suggesting this counterexample.

the example is unchanged. As before, let discount factors be close to 1. We can now repeat the argument in Example 4.3.1, noting only that the optimal strategy must condition on the states $1a$ and $1b$, which may have happened arbitrarily far in the past.

Our final consideration involves general-sum adversarial patrolling games. Both our examples above involved a kind of "punishment" meted out to the follower by the leader who plays an undesirable policy early on. APGs seem structurally quite different: the only meaningful decision by the attacker is which target to attack. The wait decision seems to have no consequences for the leader, and the leader cannot move after the attack action has been executed. Nevertheless, we now illustrate that even in this case Markov stationary policies do not suffice.

Example 4.3.3. **The leader's optimal policy in general-sum APGs may not be Markov stationary even if the only difference in utilities over targets for the attacker and defender comes from the difference in discount factors.**

Consider an APG with three targets, 1, 2, and 3. Targets 1 and 2 have value 1 and target 3 has value 0. Targets 1 and 2 are connected (to each other and each to itself); target 3 can be reached from either 1 or 2, but does not connect back to these; it is, effectively, an absorbing state. Suppose that the defender's discount factor is 0.1 and the attacker's is 0.9. That is, the defender is only concerned about what happens during the first time step, while the attacker is willing to wait. Finally, suppose that the defender starts at target 1.

Since targets 1 and 2 are identical in every way, there is an optimal Markov stationary policy that plays the same strategy in both of these. For the same reason, the probability of staying put or moving to the other valued target (1 or 2) is the same in some optimal Markov stationary policy. Let us call this probability p. Then the probability of moving to target 3 from each of these is $1-2p$; obviously, then, $p \leq 0.5$.

Since target 3 is value 0 and absorbing, clealry the attacker would attack, accruing utility of 1 (which is lost to the defender) if ever he finds the defender at that target. Suppose that the defender is at target 1 (target 2 is symmetric). Let V_A be the attacker's expected discounted value at target 1. Then the attacker will wait if and only if

$$0.9((1-2p)+2pV_A) \geq (1-p).$$

Since $V \leq 1$, we must have $1-p \leq 0.9$ or $p \geq 0.1$ to force the attacker to wait. If the attacker were to attack immediately, the expected loss to the defender is p, which is maximized when $p = 0.5$. If the attacker waits, the defender loses

$$0.1((1-2p)+2pV_D) \geq 0.1-0.2p \geq 0.08.$$

Now consider the following non-stationary policy for the defender. The defender plays $p = 0.5$ for the first two rounds, then moves to target 3 with probability 1. Clearly, the attacker will wait and attack in round 3, since his expected utility of waiting both rounds is $0.9^2 = 0.81 > 0.5$, which is what he would attain from attacking immediately. For the defender, however, the expected loss from this policy is $0.1^2 = 0.01$, much smaller than the expected loss from an optimal Markov stationary policy.

A natural question is: what makes this setting fundamentally distinct from Stochastic games, where there does, in fact, exit a Markov stationary Nash equilibrium. The first key distinction is that there could be different stationary Markov policies that are leader-optimal in different states. While true in MDPs, the following example demonstrates this assumption to be false in SSGs.

Example 4.3.4. Consider again a 3-state game with $S = 1,2,3$. Player 2 (follower) moves in state 1, player 1 (leader) moves in state 2, and state 3 is an absorbing state, just as in the examples above. Rewards are almost (but not quite) the same as well: $R_L(1,\cdot,L) = -M, R_L(1,\cdot,R) = 0; R_F(1,\cdot,L) = \varepsilon, R_F(1,\cdot,R) = 0$ and $R_L(2,D,\cdot) = \varepsilon, R_L(2,U,\cdot) = 0; R_F(2,D,\cdot) = 0, R_F(2,U,\cdot) = -M$. Transitions are a little different, however: if player 2 chooses L, the system transitions into state 2, while a move R transitions it to state 3. Once in state 2, the system always transitions to state 3. State 3 is an absorbing state in which both get 0 reward.

First, suppose that state 2 is a starting state. In this case, the optimal policy has player 1 choosing D in this state (irrelevant what he chooses in other states, or what player 2 does). Now, suppose that starting state is state 1. Then optimal policy of player 1 is to choose U in state 2, which causes player 2 to choose R in state 1.

The second distinction from MDPs and Stochastic games is that dynamic programming does not "work" in SSGs, whereas it does in stochastic games or MDPs. In fact, decisions at different time periods are intricately interdependent in SSGs, because a decision by the leader at any given time period will impact the utility and, hence, the best response of the follower both *before and after* the leader's decision point. Therefore, we cannot use backwards induction, which was the crucial step in our proof.

A natural question is whether there is any setting where a positive result is possible, besides zero-sum games where there is no distinction between Nash equilibria and SSE. Indeed, there is: team games.

Definition 4.3.1. A *team game* is a SSG with $R_L(s,a_l,a_f) = R_F(s,a_l,a_f) = R(s,a_l,a_f)$ and $\gamma_L = \gamma_F$.

Proposition 4.3.2. *For any general-sum discounted team game, there exist a leader's Markov stationary policy and a follower's deterministic Markov stationary policy that form a strong Stackelberg equilibrium. Moreover, these are both deterministic.*

Proof. Proof. Construct an MDP with the same state space as the team game, but the actions space $A = A_L \times A_F$ (which is still finite), the reward function is $R(s,a)$ where $a = (a_l,a_f) \in A$, and the transition probabilities are as in the original team game. Let π^*_{MDP} be an optimal deterministic stationary Markov policy of the resulting MDP, which is known to exist. We can decompose this policy into $\pi^*_{MDP} = (\pi^*_L, \phi^*_F)$, where the former simply specifies the leader's and the latter the follower's part in the optimal MDP policy. We now claim that (π^*_L, ϕ^*_F) constitutes a SSE.

First, we show that ϕ^*_F must be the best response to π^*_L. Let $U(\pi,\phi)$ be the expected utility of both leader and follower when following π and ϕ respectively, where expectation is taken also with respect to the initial distribution over states; that these are equal follows by the identity of the payoffs and discount factors in the team game. Note that $U(\pi,\phi) = U(\pi_{MDP} = (\pi,\phi))$, where the latter is the corresponding expected utility of the MDP we constructed above. Now, suppose that there is ϕ' which yields a higher utility to the follower. Then,

$$U(\pi^*,\phi') = U_F(\pi^*,\phi') > U_F(\pi^*,\phi^*) = U(\pi^*,\phi^*),$$

which implies that $U(\pi^*,\phi') > U(\pi^*,\phi^*)$, a contradiction, since (π^*,ϕ^*) are optimal for the MDP.

Second, we show that π^* is leader-optimal. Suppose not. Then there exists (π',ϕ') where ϕ' is a best response to π' and

$$U(\pi',\phi') = U_L(\pi',\phi') > U_L(\pi^*,\phi^*) = U(\pi^*,\phi^*),$$

which implies that $U(\pi',\phi') > U(\pi^*,\phi^*)$, a contradiction, since (π^*,ϕ^*) are optimal for the MDP. □

4.4 Computing Markov Stationary SSE

4.4.1 Exact MINLP Formulation

While in general SSE in Markov stationary strategies do not suffice, we restrict attention to these in the sequel, as general policies need not even be finitely representable. Moreover, it is unlikely that policies that are much more complicated than first-order Markov stationary would even be practically implementable. A crucial consequence of the restriction to Markov stationary strategies is that policies of the players can now be finitely represented. In the sequel, we drop the cumbersome notation and denote leader stochastic policies simply by π and follower's best response by ϕ (with π typically clear from the context). Let $\pi(a_l|s)$ denote the probability that the leader chooses $a_l \in A_L$ when he observes state $s \in S$. Similarly, let $\phi(a_f|s)$ be the probability of choosing $a_f \in A_F$ when state is $s \in S$. Above, we also observed that it suffices to focus on *deterministic* responses for the attacker. Consequently, we assume that $\phi(a_f|s) = 1$ for exactly one follower action a_f, and 0 otherwise, in every state $s \in S$.

At the root of SSE computation are the expected optimal utility functions of the leader and follower starting in state $s \in S$ defined above and denoted by $V_L(s)$ and $V_F(s)$. In the formulations below, we overload this notation to mean the variables which compute V_L and V_F in an optimal solution. Suppose that the current state is s, the leader plays a policy π, and the follower chooses action $a_f \in A_F$. The follower's expected utility is $\tilde{R}_F(s,\pi,a_f)$

$$= \sum_{a_l \in A_L} \pi(a_l|s) \left(R_F(s,a_l,a_f) + \gamma_F \sum_{s' \in S} T_{ss'}^{a_l a_f} V_F(s') \right).$$

The leader's expected utility $\tilde{R}_L(s,\pi,a_f)$ is defined analogously. Let Z be a large constant. We now present a mixed integer non-linear program (MINLP) for computing a SSE:

$$\max_{\pi,\phi,V_L,V_F} \sum_{s\in S} \beta(s) V_L(s) \tag{4.4.1a}$$

subject to :

$$\pi(a_l|s) \geq 0 \qquad \forall s, a_l \tag{4.4.1b}$$

$$\sum_{a_l} \pi(a_l|s) = 1 \qquad \forall s \tag{4.4.1c}$$

$$\phi(a_f|s) \in \{0,1\} \qquad \forall s, a_f \tag{4.4.1d}$$

$$\sum_{a_f} \phi(a_f|s) = 1 \qquad \forall s \tag{4.4.1e}$$

$$0 \leq V_F(s) - \tilde{R}_F(s,\pi,a_f) \leq (1-\phi(a_f|s))Z \; \forall s, a_f \tag{4.4.1f}$$

$$V_L(s) - \tilde{R}_L(s,\pi,a_f) \leq (1-\phi(a_f|s))Z \; \forall s, a_f \tag{4.4.1g}$$

The objective 4.4.1a of the MINLP is to maximize the expected utility of the leader with respect to the distribution of initial states. The constraints 4.4.1b and 4.4.1c simply express the fact that the leader's stochastic policy must be a valid probability distribution over actions a_l in each state s. Similarly, constraints 4.4.1d and 4.4.1e ensure that the follower's policy is deterministic, choosing exactly one action in each state s. Constraints 4.4.1f are crucial, as they are used to compute the follower best response ϕ to a leader's policy π. These constraints contain two inequalities. The first represents the requirement that the follower value $V_F(s)$ in state s maximizes his expected utility over all possible choices a_f he can make in this state. The second constraint ensures that if an action a_f is chosen by ϕ in state s, $V_F(s)$ exactly equals the follower's expected utility in that state; if $\phi(a_f|s) = 0$, on the other hand, this constraint has no force, since the right-hand-side is just a large constant. Finally, constraints 4.4.1g are used to compute the leader's expected utility, given a follower best response. Thus, when the follower chooses a_f, the constraint on the right-hand-side will bind, and the leader's utility must therefore equal the expected utility when follower plays a_f. When $\phi(a_f|s) = 0$, on the other hand, the constraint has no force.

While the MINLP gives us an exact formulation for computing SSE in general SSGs, the fact that constraints 4.4.1f and 4.4.1g are not convex, together with the integrality requirement on ϕ, make it relatively impractical, at least given state-of-the-art MINLP solution methods. Below we therefore seek a principled approximation by discretizing the leader's continuous decision space.

4.4.2 MILP Approximation

What makes the MINLP formulation above difficult is the combination of integer variables, and the non-convex interaction between continuous variables π and V_F in one case (constraints 4.4.1f), and π and V_L in another (constraints 4.4.1g). If at least one of these variables is binary, we can linearize these constraints using McCormick inequalities [79]. To enable the application of this technique, we discretize the probabilities which the leader's policy can use.

Let p_k denote a kth probability value and let $\mathscr{K} = \{1,\ldots,K\}$ be the index set of discrete probability values we use. Define binary variables $d^{a_l}_{s,k}$ which equal 1 if and only if $\pi(a_l|s) = p_k$, and 0 otherwise. We can then write $\pi(a_l|s)$ as $\pi(a_l|s) = \sum_{k\in\mathscr{K}} p_k d^{a_l}_{s,k}$ for all $s \in S$ and $a_l \in A_L$. Next, let $w^{a_l a_f}_{s,k} = d^{a_l}_{s,k} \sum_{s'\in S} T^{a_l a_f}_{ss'} V_L(s')$ for the leader, and let $z^{a_l a_f}_{s,k}$ be defined analogously for the follower. The key is that we can represent these equality constraints by the following equivalent McCormick inequalities, which we require to hold for all $s \in S$, $a_l \in A_L$, $a_f \in A_F$, and $k \in \mathscr{K}$:

$$w^{a_l a_f}_{s,k} \geq \sum_{s'\in S} T^{a_l a_f}_{ss'} V_L(s') - Z(1 - d^{a_l}_{s,k}) \tag{4.4.2a}$$

$$w^{a_l a_f}_{s,k} \leq \sum_{s'\in S} T^{a_l a_f}_{ss'} V_L(s') + Z(1 - d^{a_l}_{s,k}) \tag{4.4.2b}$$

$$-Zd^{a_l}_{s,k} \leq w^{a_l a_f}_{s,k} \leq Zd^{a_l}_{s,k}, \tag{4.4.2c}$$

and analogously for $z^{a_l a_f}_{s,k}$. Redefine follower's expected utility as $\tilde{R}_F(s,d,a_f,k) = \sum_{a_l\in A_L} \sum_{k\in\mathscr{K}} p_k \Big(R_F(s,a_l,a_f) d^{a}_{s,}$ with leader's expected utility $\tilde{R}_L(s,d,a_f,k)$ redefined similarly. The full MILP formulation is then

$$\max_{\phi, V_L, V_F, z, w, d} \sum_{s\in S} \beta(s) V_L(s) \tag{4.4.3a}$$

$$\text{subject to :}$$

$$d^{a_l}_{s,k} \in \{0,1\} \qquad \forall s, a_l, k \tag{4.4.3b}$$

$$\sum_{k\in\mathscr{K}} d^{a_l}_{s,k} = 1 \qquad \forall s, a_l \tag{4.4.3c}$$

$$\sum_{a_l\in A_L} \sum_{k} p_k d^{a_l}_{s,k} = 1 \qquad \forall s \tag{4.4.3d}$$

$$0 \leq V_F(s) - \tilde{R}_F(s,d,a_f,k) \leq (1 - \phi(a_f|s))Z \forall s, a_f \tag{4.4.3e}$$

$$V_L(s) - \tilde{R}_L(s,d,a_f,k) \leq (1 - \phi(a_f|s))Z \ \forall s, a_f \tag{4.4.3f}$$

$$\text{constraints } 4.4.1d - 4.4.1e,\ 4.4.2a - 4.4.2c.$$

Constraints 4.4.3d, 4.4.3e, and 4.4.3f are direct analogs of constraints 4.4.1c, 4.4.1f, and 4.4.1g respectively. Constraints 4.4.3c ensure that exactly one probability level $k \in \mathscr{K}$ is chosen.

4.4.3 A Bound on the Discretization Error

The MILP approximation above implicitly assumes that given a sufficiently fine discretization of the unit interval we can obtain an arbitrarily good approximation of SSE. In this section we obtain this result formally. First, we address why it is not in an obvious way related to the impact of discretization in the context of Nash equilibria. Consider a mixed Nash equilibrium s^* of an arbitrary normal form game with a utility function $u_i(\cdot)$ for each player i (extended to mixed strategies in a standard way), and suppose that we restrict players to choose a strategy that takes discrete probability values. Now, for every player i, let $\hat{s}_i$ be the closest point to s^*_i in the restricted strategy space. Since the utility function is continuous, this implies that each player's possible gain from deviating from $\hat{s}_i$ to s^*_i is small when all others play $\hat{s}_{-i}$, ensuring that finer discretizations lead to better Nash equilibrium approximation. The problem that arises in approximating an SSE is that we do not keep the follower's decision fixed when considering small changes to the leader's strategy; instead, we allow the follower to always optimally respond. In this case, the leader's expected utility can be discontinuous, since small changes in his strategy can lead to jumps in the optimal strategies of the follower if the follower is originally indifferent between multiple actions (a common artifact of SSE solutions). Thus, the proof of the discretization error bound is somewhat subtle.

First, we state the main result, which applies to all finite-action Stackelberg games, and then obtain a corollary which applies this result to our setting of discounted infinite-horizon stochastic games. Suppose that L and F are the finite sets of pure strategies of the leader and follower, respectively. Let $u_L(l,f)$ be the leader's utility function when the leader plays $l \in L$ and the follower plays $f \in F$, and suppose that X is the set of probability distributions over L (leader's mixed strategies), with $x \in X$ a particular mixed strategy with x_f the probability of playing a pure strategy $f \in F$. Let $\mathscr{P} = \{p_1,\dots,p_K\}$ and let $\varepsilon(\mathscr{P}) = \sup_{x\in X}\max_f \min_{k\in\mathscr{K}} |p_k - x_f|$. Suppose that $(x^*, f^{BR}(x^*))$ is a SSE of the Stackelberg game in which the leader can commit to an arbitrary mixed strategy $x \in X$. Let $U(x)$ be the leader's expected utility when he commits to $x \in X$.

Theorem 4.4.1. *Let $(x^{\mathscr{P}}, f^{BR}(x^{\mathscr{P}}))$ be an SSE where the leader's strategy x is restricted to $\mathscr{P}$. Then*

$$U(x^{\mathscr{P}}) \geq U(x^*) - \varepsilon(\mathscr{P}) \max_{f\in F} \sum_l |u^L(l,f)|.$$

To prove this theorem, we leverage a particular technique for computing a SSE in finite-action games: one using multiple linear programs, one for each follower strategy $f \in F$ [40]. Each of these linear programs (LP) has the general form

$$\begin{aligned}&\max_x \sum_{l\in L} x_l u^L(l,f)\\&s.t.\\&x \in \mathscr{D}(f),\end{aligned}$$

where $\mathscr{D}(f)$ is the constraint set which includes the restriction $x \in X$ and requires that the follower's choice f is his optimal response to x. To compute the SSE, one then takes the optimal solution with the best value over the LPs for all $f \in F$; the corresponding f is the follower's best response. Salient to us will be a restricted version of these LPs, where we replace $\mathscr{D}(f)$ with $\mathscr{D}^\varepsilon(f)$, where the latter requires, in addition, that leader's mixed strategies are restricted to $\mathscr{P}$ (note that $\mathscr{D}^\varepsilon(f) \subseteq \mathscr{D}(f)$). Let us use the notation $P(f)$ to refer to the linear program above, and $P^\varepsilon(f)$ to refer to the linear program with the restricted constraint set $\mathscr{D}^\varepsilon(f)$. We also use P^ε to refer to the problem of computing the SSE in the restricted, discrete, setting.

We begin rather abstractly, by considering a pair of mathematical programs, P_1 and P_2, sharing identical linear objective functions $c^T x$. Suppose that X is the set of feasible solutions to P_1, while Y is the feasible set of P_2, and $Y \subseteq X \subseteq \mathbb{R}^m$. Let OPT_1 be the optimal value of P_1.

Lemma 4.4.2. *Suppose that $\forall x \in X$ there is $y \in Y$ such that $\|x-y\|_\infty \leq \varepsilon$. Let $\hat{x}$ be an optimal solution to P_2. Then $\hat{x}$ is feasible for P_1 and $c^T\hat{x} \geq OPT_1 - \varepsilon \sum_i |c_i|$.*

Proof. Proof. Feasibility is trivial since $Y \subseteq X$. Consider an arbitrary optimal solution x^* of P_1. Let $\tilde{x} \in Y$ be such that $\|x^* - \tilde{x}\|_\infty \leq \varepsilon$; such $\tilde{x}$ must exist by the condition in the statement of the lemma. Then

$$\begin{aligned} c^T x^* - c^T \tilde{x} &= \sum_i c_i(x_i^* - \tilde{x}_i) \leq |\sum_i c_i(x_i^* - \tilde{x}_i)| \\ &\leq \sum_i |c_i||x_i^* - \tilde{x}_i| \leq \varepsilon \sum_i |c_i|, \end{aligned}$$

where the last inequality comes from $\|x^* - \tilde{x}\|_\infty \leq \varepsilon$. Finally, since $\hat{x}$ is an optimal solution of P_2 and $\tilde{x}$ is P_2 feasible, $c^T \hat{x} \geq c^T \tilde{x} \geq c^T x^* - \varepsilon \sum_i |c_i| = OPT_1 - \varepsilon \sum_i |c_i|$. □

We can apply this Lemma directly to show that for a given follower action f, solutions to the corresponding linear program with discrete commitment, P_f^ε, become arbitrarily close to optimal solutions (in terms of objective value) of the unrestricted program P_f.

Corollary 4.4.3. *Let $OPT(f)$ be the optimal value of $P(f)$. Suppose that $x^\varepsilon(f)$ is an optimal solution to $P^\varepsilon(f)$. Then x^ε is feasible in $P(f)$ and*

$$\sum_{l \in L} x_l^\varepsilon u^L(l, f) \geq OPT(f) - \varepsilon \sum_l |u^L(l, f)|.$$

We now have all the necessary building blocks for the proof.

Proof. Proof of Theorem 4.4.1 Let $\hat{x}$ be a SSE strategy for the leader in the restricted, discrete, version of the Stackelberg commitment problem, P^ε. Let x^* be the leader's SSE strategy in the unrestricted Stackelberg game and let f^* be the corresponding optimal action for the follower (equivalently, the corresponding $P(f)$ which x^* solves). Letting $\hat{x}^{f^*}$ be the optimal solution to the restricted LP $P(f^*)^\varepsilon$, we apply Corollary 4.4.3 to get

$$\begin{aligned} \sum_{l \in L} \hat{x}^{f^*} u^L(l, f^*) &\geq OPT(f) - \varepsilon \sum_l |u^L(l, f^*)| \\ &= U(x^*) - \varepsilon \sum_l |u^L(l, f^*)|, \end{aligned}$$

where the last equality is due to the fact that x^* is both an optimal solution to Stackelberg commitment, and an optimal solution to $P(f^*)$.

Since $\hat{x}$ is optimal for the restricted commitment problem, and letting $\hat{f}$ be the corresponding follower strategy,

$$\begin{aligned} U(\hat{x}) = \sum_{l \in L} \hat{x}_l u^L(l, \hat{f}) &\geq \sum_{l \in L} \hat{x}^{f^*} u^L(l, f^*) \\ &\geq U(x^*) - \varepsilon \sum_l |u^L(l, f^*)| \\ &\geq U(x^*) - \varepsilon \max_{f \in F} \sum_l |u^L(l, f)|. \end{aligned}$$

□

The result in Theorem 4.4.1 pertains to general *finite-action* Stackelberg games. Here, we are interested in SSGs, where pure strategies of the leader and follower have, in general, arbitrarily infinite sequences of decisions. However, if we restrict attention to Markov stationary policies for the leader, we guarantee that the consideration set of the leader is finite, allowing us to apply Theorem 4.4.1.

Corollary 4.4.4. *In any SSG in which the leader is restricted to Markov stationary policies, the leader's expected utility in a SSE can be approximated arbitrarily well using discretized policies.*

4.4.4 Comparison Between MINLP and MILP

Above we asserted that the MINLP formulation is likely intractable given state-of-the-art solvers as motivation for introducing a discretized MILP approximation. We now support this assertion experimentally.

For the experimental comparison between the two formulations, we generate random stochastic games as follows. We fix the number of leader and follower actions to 2 per state and the discount factors to $\gamma_L = \gamma_F = 0.95$. We also restricted the payoffs of both players to depend only on state $s \in S$, but otherwise generated them uniformly at random from the unit interval, i.i.d. for each player and state. Moreover, we generated the transition function by first restricting state transitions to be non-zero on a predefined graph between states, and generated an edge from each s to another s' with probability $p = 0.6$. Conditional on there being an edge from s to s', the transition probability for each action tuple (a_l, a_f) was chosen uniformly at random from the unit interval.

	Exp Utility	Running Time (s)
MINLP (5 states)	9.83	375.26
MILP (5 states)	10.16	5.28
MINLP (6 states)	9.64	1963.53
MILP (6 states)	11.26	24.85

Table 4.1. Comparison between MINLP and MILP ($K = 5$), based on 100 random problem instances.

Table 4.1 compares the MILP formulation (solved using CPLEX) and MINLP (solved using KNITRO with 10 random restarts). The contrast is quite stark. First, even though MILP offers only an approximate solution, the actual solutions it produces are *better* than those that a state-of-the-art solver gets using MINLP. Moreover, MILP (using CPLEX) is more than 70 times faster when there are 5 states and nearly 80 times faster with 6 states. Finally, while MILP solved every instance generated, MINLP successfully found a feasible solution in only 80% of instances.

4.5 Extended Example: Patrolling the Newark Bay and New York Harbor

Consider again the example of patrolling the Newark Bay and New York Harbor under the geographic constraints shown in Figure 4.1. We now study the structure of defense policies in a variant of this patrolling example problem that are both deviations from zero-sum games. Our examples are motivated by some basic reasons for the significance of departure from zero-sum games in security settings, despite the fact that interests of players are clearly adversarial. In both variants we therefore assume that the actual values of targets to both players are identical and as shown in the figure.

In the first example, the sole departure from strict competitiveness is in allowing the defender and attacker to disagree about the way they discount future payoffs. Specifically, keeping everything else equal, we systematically vary γ_L and γ_F. Figure 4.3 shows the most relevant portion of the defender's policy for the cross-product of three values for γ_L and γ_F: 0.1, corresponding to an extremely impatient player, 0.75, a moderate level of patience, and 0.999, a nearly extreme level of patience. In this figure, as well as the one below, the thickness of an edge roughly corresponds to the probability of the associated defense move.

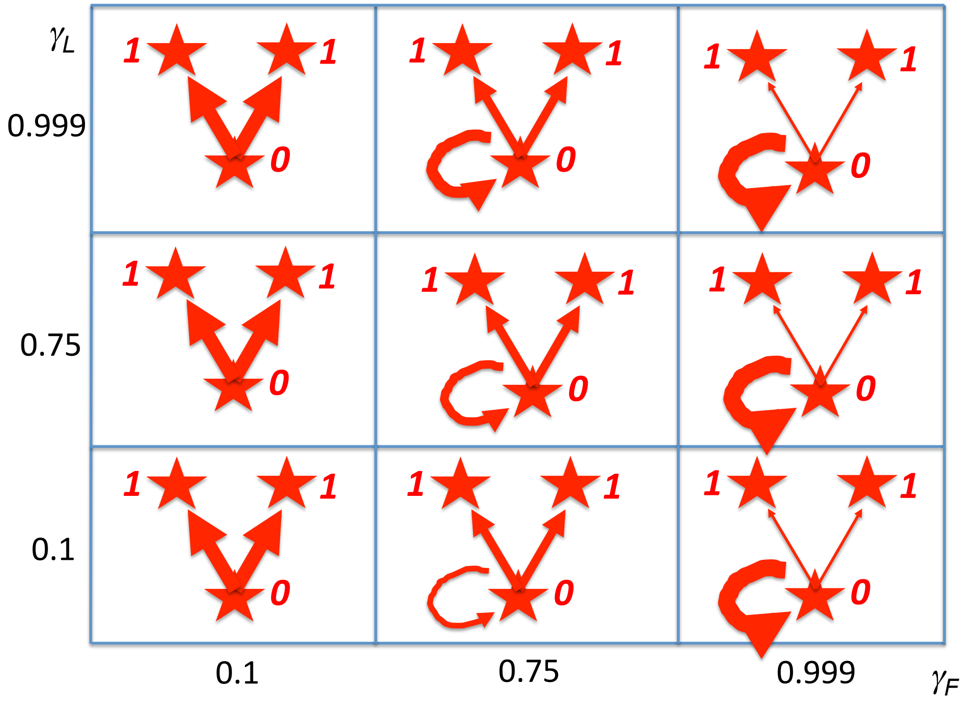


Figure 4.3. Varying the discount factors γ_L and γ_F.

We can observe two important patterns. The first is that the *defender's* discount factor plays little role in determining his policy. The second is that as the attacker becomes increasingly patient, the defender spends more time at base (the bottom target), even though it has no value to either. This last result may seem quite surprising at first. Recall from Figure 4.1, however, that the base is connected to both of the high-value targets, but these are not connected to each other. As soon as the defender commits to one of these, the attacker obtains the highest payoff by attacking the other. The defender will therefore profit by keeping the attacker guessing as long as possible, staying at base, but always with a threat to cover a high-value target.

Our second example maintains the zero-sum assumption on payoffs, and even lets the discount factors be identical for both players. This example is motivated by a basic reason for the significance of departure from zero-sum games in security settings, despite the fact that interests of players are clearly adversarial: we assume that the actual values of targets to both players are identical and as shown in the figure. The departure from strict competitiveness comes from allowing the attacker (but not the defender) to be risk averse.

To model risk aversion, we filter the payoffs through the exponential function $f(u) = 1 - e^{-\alpha u}$, where u is the original payoff. This function is well known to uniquely satisfy the property of constant absolute risk aversion (CARA) [56]. The lone parameter, α, controls the degree of risk aversion, with higher α implying more risk averse preferences.

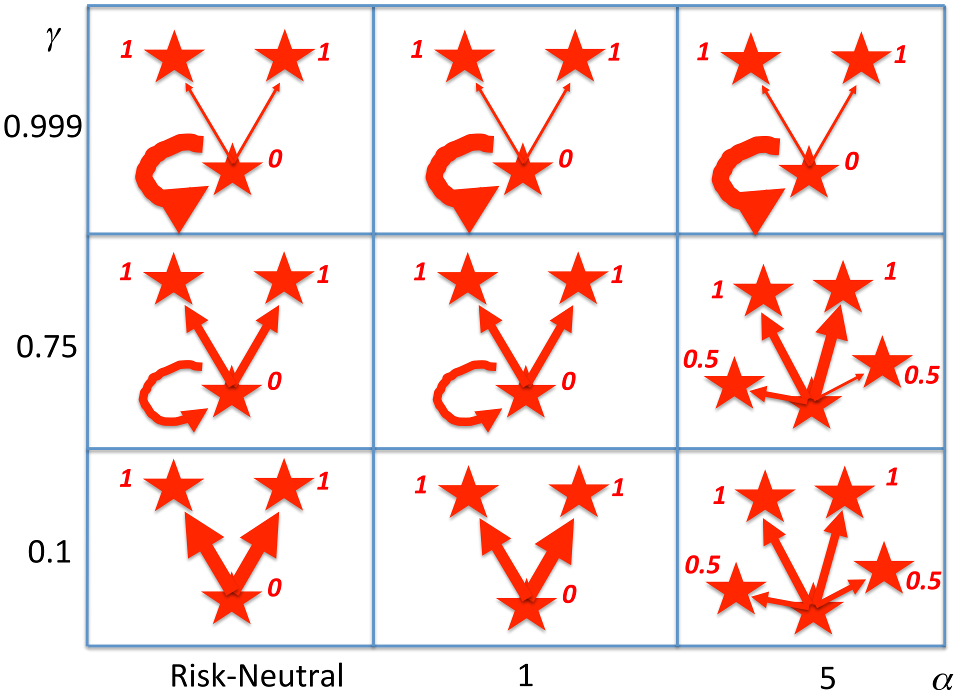


Figure 4.4. Varying discount factors $\gamma = \gamma_L = \gamma_F$ and the degree of risk aversion α.

In Figure 4.4 we report the relevant portion of the defense policy in the cross-product space of three discount factor values (0.1, 0.75, and 0.999) and three values of risk aversion (risk neutral, and $\alpha = 1$ and 5). We can make two qualitative observations. First, as the attacker becomes increasingly risk averse, the entropy of the defender's policy increases (i.e., the defender patrols a greater number of targets with positive probability). This observation is quite intuitive: if the attacker is risk averse, the defender can profitably increase the attacker's uncertainty, even beyond what would be optimal with a risk neutral attacker. Second, the impact of risk aversion diminishes as the players become increasingly patient. This is simply because a patient attacker is willing to wait a longer time before an attack, biding his time until the defender commits to one of the two most valued targets; this in turn reduces his exposure to risk, since he will wait to attack only when it is safe.

4.6 Optimal Adversarial Patrolling on Networks

Adversarial patrolling games are an important special case of SSGs; indeed, these provide, perhaps, the best practical motivation for studying SSGs. We begin by considering the problem of computing Markov stationary Stackelberg equilibria in general-sum APGs (specializing the corresponding formulation for SSGs), and then proceed to focus on a further specialization to zero-sum APGs, for which we can obtain far more scalable formulations.

4.6.1 A MINLP Formulation for General-Sum APGs

While we have already provided a formulation for computing Markov stationary policies in general SSGs, we now offer a specialized formulation for APGs which is somewhat (though not very much) more scalable than the general version.

$$\max_{\pi, v^a, v^d, a, b} \; v_0^d \tag{4.6.1a}$$
$$\pi_{ij} \geq 0 \tag{4.6.1b}$$
$$\sum_j \pi_j = 1 \tag{4.6.1c}$$
$$\pi_{ij} \leq A_{ij} \tag{4.6.1d}$$
$$b_i \in \{0, 1\} \tag{4.6.1e}$$
$$0 \leq v_i^a - R_i^a \leq b_i Z \tag{4.6.1f}$$
$$0 \leq v_i^a - \gamma_a \sum_j \pi_{ij} v_j^a \leq (1 - b_i) Z \tag{4.6.1g}$$
$$v_i^d - R_i^d \leq b_i Z \tag{4.6.1h}$$
$$v_i^d - \gamma_d \sum_j \pi_{ij} v_j^d \leq (1 - b_i) Z \tag{4.6.1i}$$
$$a_{ij} \in \{0, 1\} \tag{4.6.1j}$$
$$\sum_j a_{ij} = 1 \tag{4.6.1k}$$
$$0 \leq R_i^a - (1 - \pi_{ij}) U_a^u(j) - \pi_{ij} U_a^c(j) \leq (1 - a_{ij}) Z \tag{4.6.1l}$$
$$R_i^d - (1 - \pi_{ij}) U_d^u(j) - \pi_{ij} U_d^c(j) \leq (1 - a_{ij}) Z. \tag{4.6.1m}$$

In the MINLP 4.6.1, b_i is an integer variable used to determine whether the attacker attacks or waits upon seeing state i, and a_{ij} are integer variables that determine which target the attacker would attack if he chose to do so upon seeing state i. Constraints 4.6.1f and 4.6.1g correspond to attacker utility of attacking and waiting in state i, respectively. These compute the expected value of the attacker in state i. Constraints 4.6.1h and 4.6.1i subsequently compute the defender value in state i using attacker decision b_i. Constraint 4.6.1l computes which target is attacked by the attacker (if he chooses to do so) in state i, and the corresponding attacker expected utility R_i^a. Constraint 4.6.1m computes expected defender utility if attacker chooses to attack in state i.

Since the MINLP for general-sum APGs is non-convex, it is clearly impractical, and, just as before, we can obtain a MILP approximation by discretizing the probabilities, just as we had done above. While certain settings truly warrant a general-sum model, however, in adversarial situations it is quite natural to consider a zero-sum restriction, which we do next.

4.6.2 Zero-Sum APGs: A Baseline Formulation

Since an APG is a special case of a stochastic game, and since a Stackelberg equilibrium is equivalent to a Nash equilibrium in zero-sum games, we can directly lift the bilinear programming formulation for computing Nash equilibria for two-player zero-sum games from [49], specializing it to our setting. One minor change to their formulation that becomes crucial as we consider alternative models below is to represent the constraints on the defender's action imposed by the graph

G as a set of linear constraints in the formulation. Note also that in a zero-sum formulation, the discount factors of both the attacker and defender must be identical. We let $\delta = \gamma_a = \gamma_d$ be the common discount factor. Recalling that v_i represents (and, in this case, computes) the expected attacker value of starting in state i, we can formulate the defender's problem as the following bilinear program:

$$\begin{aligned}
&\min_{\pi,v} \sum_i v_i && && (4.6.2a)\\
&\text{s.t.}: && && \\
&\pi_{ij} \geq 0 && \forall\, i,j \in T && (4.6.2b)\\
&\sum_j \pi_{ij} = 1 && \forall\, i \in T && (4.6.2c)\\
&\pi_{ij} \leq A_{ij} && \forall\, i,j \in T && (4.6.2d)\\
&v_i \geq (1-\pi_{ij})U_a^u(j) + \pi_{ij}U_a^c(j) && \forall\, i,j \in T && (4.6.2e)\\
&v_i \geq \delta \sum_j \pi_{ij} v_j && \forall i \in T. && (4.6.2f)
\end{aligned}$$

Constraints 4.6.2b and 4.6.2c simply constrain defender policy to be a valid probability distribution, and constraint 4.6.2d restricts that defender's moves must obey the specified graph. The key constraints 4.6.2e and 4.6.2f are easiest to think about if we fix defender policy π and just consider the MDP faced by the attacker. The right-hand-side of Constraint 4.6.2e corresponds to the expected utility of attacking immediately, while the right-hand-side of Constraint 4.6.2f is the expected value of waiting (immediate reward is 0 for a waiting action). The constraints then arise because the state v_i must be the expected utility of making the best action choice, and minimizing the objective ensures that these values bind to *some* action in every state.

An important observation about the NLP formulation is that it only involves n non-linear constraints, far fewer than a NLP formulation to compute equilibria in general zero-sum stochastic games. As we demonstrate below, this NLP therefore scales extremely well with the number of targets, in large part because, additionally, every local optimum is a global optimum [49]. Nevertheless, we note that we can, again, transform this problem into a MILP approximation by discretizing the defense probabilities.

4.6.3 Extensions to the Basic Model

We now consider several extensions to the basic model that capture several elements that reflect realistic patrolling settings, but are not at the moment captured. In what follows, we restrict attention to zero-sum APG settings, although analogous extensions can be directly lifted to the general-sum case as well.

Defender with Multiple Resources

Our treatment of APGs thus far assumed that the defender has only a single resource to patrol with (e.g., USCG has only one boat). We now show that our formulation can be naturally generalized to allow the defender an arbitrary number of resources. First, rather than working with targets directly, we must work with coverage vectors, which we now identify with states of the corresponding stochastic game. Thus, a state (or coverage vector) s is a binary vector with $s_i = 1$ if and only if target i is covered by the defender. We let r be the number of defender resources, so that any valid s has at most r 1's; we let S denote the set of all valid states (coverage vectors). The attacker can observe the current coverage vector s of the defender, while the defender's decision is the probability $\pi_{ss'}$ of moving to a new coverage vector s' starting at s. We also let v_s be the expected discounted attacker utility when he observes the coverage vector s. As before, the graph with adjacency matrix A constrains defender moves between targets and, consequently, there are constraints between feasible moves between states induced by A. Let $B_{ss'} = 1$ if and only if s' is a valid transition for the defender starting with a coverage vector s. Below, we show how to obtain this matrix given A. Finally, attacker utility function must now be defined with respect to coverage vectors s, as well as attacked targets j. Let $U_a(s, j)$ be the utility the attacker obtains from attacking j if the defense coverage vector is s. The full NLP formulation of the resulting problem is:

$$\min_{\pi,v} \sum_s v_s \tag{4.6.3a}$$

$$\text{s.t. :}$$

$$\pi_{ss'} \geq 0 \qquad \forall\, s, s' \in S \tag{4.6.3b}$$

$$\sum_{s'} \pi_{ss'} = 1 \qquad \forall\, s \in S \tag{4.6.3c}$$

$$\pi_{ss'} \leq B_{ss'} \qquad \forall\, s, s' \in S \tag{4.6.3d}$$

$$v_s \geq \sum_{s'} \pi_{ss'} U_a(s', j) \qquad \forall\, s \in S, j \in T. \tag{4.6.3e}$$

$$v_s \geq \delta \sum_{s'} \pi_{ss'} v_{s'} \qquad \forall s \in S. \tag{4.6.3f}$$

As we can see, the NLP 4.6.3 is quite similar to the formulation we introduced above. The key differences are the derivation of the state adjacency matrix B from target adjacencies A, as well as utilities $U_a(s, j)$ based on $U_a^c(j)$ and $U_a^u(j)$. Since the latter question is simpler, let us tackle it first. Suppose that s is a coverage vector and attacker attacks target j. Then $U_a(s, j) = U_a^c(j)$ if $s_j = 1$ and $U_a(s, j) = U_a^u(j)$ otherwise. To obtain the matrix B, consider a pair of coverage vectors s and s', and let T_s be the set of covered targets under s and $T_{s'}$ the set of targets covered under s'. Next, let $G_{ss'} = \{T_s, T_{s'}, E\}$ be a bipartite graph with a directed edge $(i, j) \in E$ if and only if $i \in T_s$, $j \in T_{s'}$, and $A_{ij} = 1$. The following proposition is then relatively direct.

Proposition 4.6.1. *For each $s, s' \in S$, $B_{ss'} = 1$ if and only if $G_{ss'}$ has a perfect matching.*

Proof. Proof. For one direction, suppose that $G_{ss'}$ has a perfect matching. This means that every covered target in s is matched to exactly one covered target in s', which implies that there is a feasible move for a resource situated at each target covered under s to s' and, since this is a matching, no two resources move to the same location. For the other direction, suppose that the maximum matching is not a perfect matching. Since this matching matches the largest number of covered targets, it must be that under every possible matching there exists an infeasible move for some resource. □

The convenience of this result is that the existence of a perfect matching can be checked in time polynomial in the number of resources r [47].

Our use of coverage vectors when the defender has multiple resources contrasts with the approach offered by [21] in a similar context for undiscounted games, who instead identify "states" with vectors $c = \{c_1, \ldots, c_r\}$, where c_d specifies the target covered by a defender d. Observe that the set all possible states c in the Basilico et al. formulation is therefore the set of all r-length permutations of targets. In contrast, our formulation only considers r-length *combinations*. Consequently, each coverage vector s corresponds to $r!$ distinct states c. Our formulation is therefore exponentially more compact.

Attacks Taking Multiple Time Steps

An important assumption in the baseline formulation above is that once the attacker chooses to attack, the actual attack commences on the next time step. We now consider a generalization in which attacks take an arbitrary number of steps $h \geq 0$ to unfold. For clarity, we extend only the baseline model here, assuming that the defender has only a single defense resource. Observe that stationarity in the space of targets no longer suffices in general. To see this, consider a simple example with 3 targets (1, 2, and 3), and suppose that there are edges between 1 and 2, and between 2 and 3 only. Finally, suppose that $h = 4$ and the attacker is infinitely patient. Then a nonstationary policy in which the defender moves from 1 to 2 to 3 to 2 to 1 is optimal, since the attacker will always be caught. On the other hand, no stationary policy exists which guarantees that we always catch the attacker: if the defender is at target 2, and the policy is deterministic, he will necessarily leave some target uncovered; if the policy at 2 is stochastic, there is strictly positive probability that the attacker will not be caught.[2]

That it no longer suffices to consider policies which only condition on the previous defender move is unfortunate: keeping track of h-step histories, which is now required, means that our formulations will have $O((n+1)^h)$ states (recall that n is the number of targets), becoming intractable even when n and h are relatively small. A natural question is therefore whether we can obtain good approximations considering relatively short histories of length $K < h$. We offer below a formulation which can be tuned using an arbitrary choice of K, for a fixed (and given) h.

Let $s = \{i_1, \ldots, i_K\}$ be a sequence of K defender moves, and let $\pi_{s,j}$ denote the probability of moving to j given "current state" s, that is, given the fact that the defender previously followed a sequence s. Let S be the set of all *feasible* sequences of moves of length K (that is, sequences,

[2] We thank Zhengyu Yin for suggesting this example.

such that $A_{i_k,i_{k+1}} = 1$ for all k). Similarly, expected value of the attacker is now a function of state, and we denote it by v_s. Let M be a three-dimensional matrix, with $M_{s,j,s'} = 1$ iff moving to j after a history vector s results in a new history s'. Finally, let $A_{sj} = 1$ iff it is feasible for the defender to move to j when his previous sequence was s. A_{sj} can be computed simply by checking that $A_{i_K j} = 1$, where i_K is the last visited target in s. The full NLP formulation for this problem is then

$$\min_{\pi,v,\alpha} \sum_s v_s \tag{4.6.4a}$$

s.t. :

$$\pi_{s,j} \geq 0 \qquad \forall\, j \in T, s \in S \tag{4.6.4b}$$

$$\sum_j \pi_{s,j} = 1 \qquad \forall\, j \in T, s \in S \tag{4.6.4c}$$

$$\pi_{s,j} \leq A_{ij} \qquad \forall\, j \in T, s \in S \tag{4.6.4d}$$

$$\alpha^1_{s,j} = \pi_{s,j} \qquad \forall j \in T, s \in S \tag{4.6.4e}$$

$$\alpha^t_{s,j} = \sum_{k \neq j} \pi_{s,k} \sum_{z \in S} M_{s,k,z} \alpha^{t-1}_{z,j} \qquad \forall\, j \in T, s \in S, t \leq h \tag{4.6.4f}$$

$$v_s \geq (1 - \sum_{t=1}^{h} \alpha^t_{s,j}) U^u_j + \sum_{t=1}^{h} \alpha^t_{s,j} U^c_j \qquad \forall\, j \in T, s \in S \tag{4.6.4g}$$

$$v_s \geq \delta \sum_j \pi_{s,j} \sum_{z \in S} M_{s,j,z} v_z \qquad \forall j \in T, s \in S. \tag{4.6.4h}$$

Here, $\alpha^t_{s,,j}$ is the probability that a target j will be visited by the defender in exactly t time steps without passing through j in the process, given history s; it is computed using Constraints 4.6.4e and 4.6.4f. Constraints 4.6.4g compute the attacker's utility if he chooses to attack, while Constraints 4.6.4h compute the expected utility of waiting.

4.7 Experiments: Patrolling on Exogenous Graphs

In our experimental studies below we use a somewhat simplified model in which $U^c_a(i) = 0$ for all targets $i \in T$. We generate the values of successful attacks $U^u_a(i)$ i.i.d. from a uniform distribution on a unit interval. Throughout, we use $\delta = 0.95$, except where specified otherwise.[3] We use well-known generative models for networks to generate random instances of graphs over which the defender patrols. The first is an Erdos-Renyi model [84] under which every directed link is made with a specified and fixed probability p; we refer to this model by ER(p), or simply ER. The second is Preferential Attachment [84], which adds nodes in a fixed sequence, starting from an arbitrary seed graph with at least two vertices. Each node i is attached to m others stochastically (unless $i \leq m$, in which case it is connected to all preceding nodes), with probability of connecting to a node j proportional to the degree of j, d_j. In a generalized version of this model that we

[3] We considered other discount factors as well, but this one strikes the right balance: it creates interesting tradeoffs between attacking and waiting, and yet creates a setting that is significantly different from past work which only considers $\delta = 1$. We study the impact of the discount factor in Section 4.7.5.

consider below, connection probabilities are $(d_j)^\gamma$, such that when $\gamma = 0$ we recover (roughly) the Erdos-Renyi model, $\gamma = 1$ recovers the "standard" PA model, and large values of γ correspond to highly inhomogeneous degree distributions. Finally, we also consider simple Cycles.

When the networks are relatively sparse (like a Cycle), and the number of targets large, the attacker can usually attack the most valuable target at time 0, and not face the tradeoff between the value of time and attack utility that we are trying to model. In our experiments, we therefore connected the starting target 0 to every other target, with network topology effective only on the rest of the targets. We may think of target 0 as a base, and the rest of the targets as initial deployments, which are unconstrained. Since target 0 is only a nominal target, we additionally set its utility to the attacker $U_a^u(0)$ to be 0.

All computational experiments were performed on a 64 bit Linux 2.6.18-164.el5 computer with 96 GB of RAM and two quad-core hyperthreaded Intel Xeon 2.93 GHz processors. We did not make use of any parallel or multi-threading capabilities, restricting a solver to a single thread, when relevant. Mixed integer linear programs were solved using CPLEX version 12.2, mixed integer non-linear programs were solved using KNITRO version 7.0.0, and we used IPOPT version 3.9.3 to solve non-linear (non-integer) programs in most cases (the one exception is identified below, where we also used KNITRO).

The results we report are based on 100 samples from both the attacker utility distribution and (when applicable) from the network generation model. Throughout, we report 95% confidence intervals, where relevant.

4.7.1 Comparison to Basilico et al.

[19] presented a multiple math programming approach to adversarial patrolling for a setting very similar to ours. By setting $\delta = 1$, and reformulating the algorithm in Basilico et al. in a zero-sum setting and with a single-step attack, we can make a direct comparison between our algorithm (using the NLP formulation) and theirs. The results, shown in Figure 4.5 (left), suggest that our approach yields significantly better solutions. The difference becomes less important as the number of targets increases: since in both approaches we only allow for one defender resource (defender can protect at most a single target at a time), and we assign relative values to targets uniformly randomly, on sparse graphs the attacker becomes increasingly likely to get the target he wants when the discount factor is 1, since the defender is eventually at least two hops away from the most valuable target.

It may be quite puzzling that, in a sense, our approach yields solutions better to Basilico et al., even "playing on their turf", that is, having an attacker that is infinitely patient. We now proceed to show specifically why the approach offered by Basilico et al. is suboptimal; to our knowledge, we are the first to offer this analysis of what is currently the state-of-the-art (all the current approaches build on the same core framework).

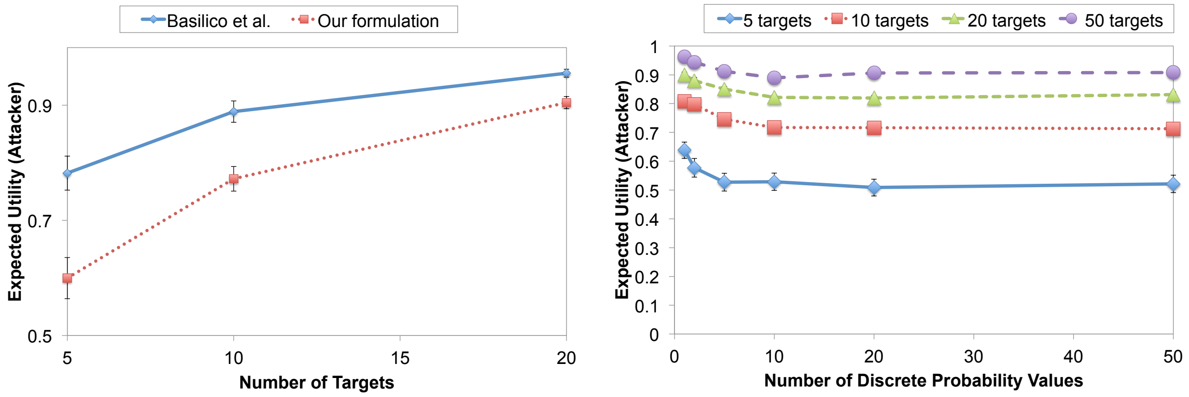


Figure 4.5. Left: Comparison between our NLP formulation and that developed by Basilico et al. Right: MILP objective value as a function of granularity of discretization. The graph is ER(0.1) in both cases.

Suboptimality of Basilico et al.

Suboptimality of Attacker Policies. One crucial assumption made by [19] and subsequent papers is that when attacker does not discount rewards (i.e., $\delta = 1$) attacker policies can take the compact form of *enter_when(j,i)*, meaning that the attack commences if and only if the defender is observed at target *i*, in which case *j* is attacked, and wait otherwise. We now demonstrate that this restriction is, in general, suboptimal for the attacker. Consider Figure 4.6. The labels on nodes are node numbers. The labels over the directed edges correspond to defender transition probabilities (based on the defender policy, constrained by the underlying graph). The number next to a node corresponds to its value (loss to the defender if that node is successfully attacked).

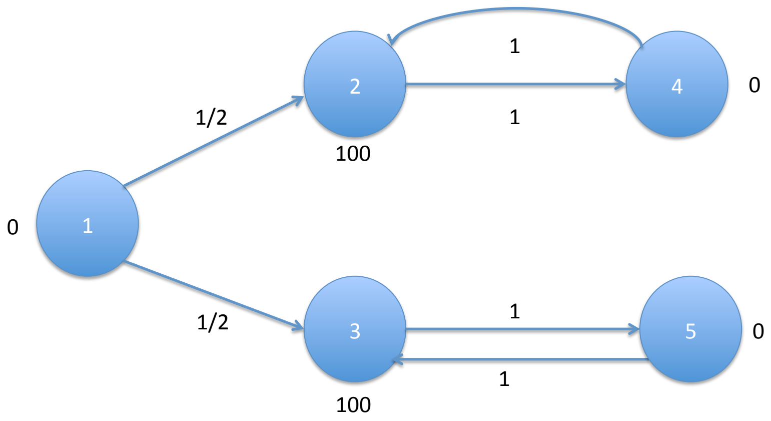


Figure 4.6. Example of suboptimality of restricted attacker policies.

Now, consider the optimal unrestricted attacker policy, assuming it takes a single step for the attacker to attack. Since nodes 2 and 3 have the highest value and are not directly connected, the attacker will attain a utility of 100 with probability 1 by attacking 2 if the defender is at 3, attacking 3 if the defender is observed at 2, and waiting otherwise. Next, suppose we restrict the attacker to a policy of the form *enter_when(j,i)*, thereby restricting him to attack only upon seeing the defender at a particular target, and no other. Clearly, it suffices to attack either 2 or 3, so we can easily enumerate all possibilities here. It is then easy to see that the expected utility of the attacker, assuming the defender starts at target 1 is 50 no matter which defender location triggers an attack. Thus, optimal attacker policy is twice the approximation. In fact, it is easy to see that we can extend this example to make the approximation ration arbitrarily large (and $O(n)$).

Suboptimality of Defender Policies. The proposed approach by Basilico et al. at computing optimal Stackelberg commitment in adversarial patrolling settings is by making use of multiple NLPs, each for a specific *enter_when(j,i)* strategy by the attacker. In each NLP, a set of constraints are imposed that the *enter_when(j,i)* has a higher utility than all n^2 *enter_when(j′,i′)* alternatives. We now demonstrate that this approach is suboptimal. At the intuitive level, what is missing is the fact that some alternatives considered in a given NLP may actually not be reachable given the initial distribution over defended targets. Consequently, the NLPs may be overconstrained and, at times, appear infeasible.

To begin, assume that we have a zero-sum game, and attacks take a single time step to unfold. Suppose that the defender starts at target 1. Further, assume that the attacker gets 0 if he waits or gets caught attacking, and the value of an attacked target if he does not get caught. Moreover, assume that attacker optimal policies are of the form *enter_when(j,i)* (ignoring the complications we identified in the previous section). One fortunate aspect of these assumptions is that now every NLP formulated by Basilico et al. reduces to a linear program. Specifically, suppose that we are considering an optimal decision for the defender under the constraint that attacker follows a strategy *enter_when(r,s)* for a specific r and s. Using our notation defined above, the linear program optimizing defender's policy then becomes

$$
\begin{aligned}
&\min_{\pi}(1-\pi_{s,r})u_r && \text{s.t.}: & \\
&\pi_{ij} \geq 0 && \forall\, i,j \in T & \text{(4.7.1a)}\\
&\sum_j \pi_{ij} = 1 && \forall\, i \in T & \text{(4.7.1b)}\\
&\pi_{ij} \leq A_{ij} && \forall\, i,j \in T & \text{(4.7.1c)}\\
&(1-\pi_{s,r})u_r \geq (1-\pi_{z,w})u_w && \forall\, z,w \in T. & \text{(4.7.1d)}\\
& && & \text{(4.7.1e)}
\end{aligned}
$$

We are primarily interested here in Constraints 4.7.1d, which are intended to represent attacker's preference for *enter_when(r,s)* compared to other alternatives, as in standard multiple-LP approaches to solving one-shot Stackelberg games. The main problem with these constraints as formulated is that they do not correctly compute expected utility accounting for both the initial distribution of defended over the targets, as well as defender's policy. As we are about to show, this results in an overconstrained problem.

Consider the example in Figure 4.7. This figure shows constraints on the defender policies (the graph), as well as the values of each node (numbers next to nodes). Node labels are just identifiers of targets. First, we observe that the optimal strategy of the defender yields the attacker expected utility of 0.85. Consider the defender strategy with $\pi_{1,2} = \pi_{1,4} = 0$, $\pi_{1,3} = 1$, and $\pi_{3,1} = 0$, the attacker's best response is to attack either 2 or 4, and the attacker's utility is 0.85. Now, note that putting positive probability on either 2 or 4 cannot be a part of an optimal defender strategy. If we suppose that $\pi_{1,2} > 0$, for example, the attacker's expected utility will clearly be strictly greater than 0.85, since with this probability he will attack the more valuable target. Thus, an optimal defense policy would have $\pi_{1,2} = \pi_{1,4} = 0$. It then follows that the previous defender policy is in fact optimal, with attacker utility 0.85.

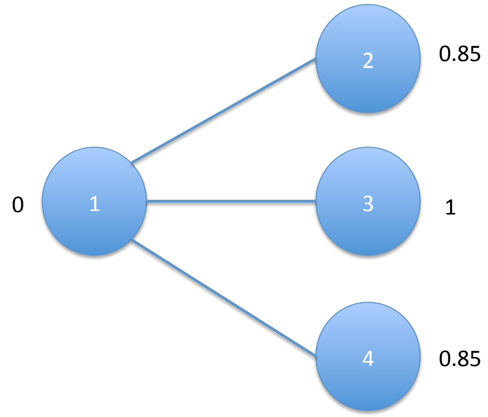


Figure 4.7. Example of suboptimality of defender policies in the Basilico et al. multiple-NLP formulation.

Now, consider an LP solved for the attacker strategy *enter_when(*2*,*1*)*. This means that Constraints 4.7.1d become

$$0.85(1-\pi_{1,2}) \geq u_w(1-\pi_{z,w}) \quad \forall\, z, w.$$

Next, consider $z = 4$ and $w = 3$. Since there is no edge between these targets, and, consequently, when the defender is at target 4 he cannot visit 3 in a single step, the constraint becomes

$$0.85(1-\pi_{1,2}) \geq 1.$$

Since $1 > 0.85$, this constraint, and therefore the resulting LP, is infeasible. Note, however, that if $\pi_{1,4} = 0$, this constraint is in fact irrelevant, as target 4 will never be reached by the defender! In an optimal formulation, this constraint should therefore not be present. Precisely the same argument can be made for any LP where attacker strategy involves attacking either target 2 or 4. Consequently, the only feasible LPs will involve the attacker attacking target 3. Moreover, since $z = 4$ and $w = 3$ are always available alternatives, it must be that the attacker's expected utility for any optimal policy of a feasible LP is 1, which is higher than the optimal utility 0.85. Notice that by appropriately scaling the node values, we can make this difference arbitrarily large.

In a recent paper, [20] attempted to eliminate the problem noted here by checking whether the resulting "optimal" policy is consistent, in the sense that the observed target which triggers an attack, that is, x in an attacker's best response *enter_when(t,x)*, must be reachable if the defender follows this policy. Their meta-algorithm then solves the multiple-NLP formulation for different subsets of targets, and returns the optimal consistent solution. We now demonstrate that our counterexample yields a consistent solution and, consequently, even the meta-algorithm of [20] is still arbitrarily suboptimal.

Consider one particular "optimal" policy solved for the above example: the one to which the attacker's best response is *enter_when(*$3,1$*)*. One of the constraints of the corresponding LP is

$$(1-\pi_{1,3}) \geq (1-\pi_{4,3}) = 1,$$

which implies that $\pi_{1,3} = 0$ in an "optimal" policy. The key observation is that this policy is consistent, since target 1 is visited by the defender w.p. 1, as it is the defender's starting point.

4.7.2 MILP Discretization

The size and, consequently, complexity of the MILP depends greatly on the fineness of discretization of the probability interval. While we can, perhaps, presume that a fine enough discretization would get us close to an optimal solution, computationally we cannot in all likelihood afford a very fine discretization. An important question, therefore, is: how much is enough? We address this question by considering a sequence of increasingly fine discretizations, starting at $L = 1$ ($p_0 = 0$ and $p_1 = 1$) and going up to $L = 50$ ($p_l \in \{0, 0.02, 0.04, \ldots, 1\}$). To ensure that whatever we find is not particular to a given setting, we also vary the number of targets between 5 and 50, as well as the network topology (Cycle, Erdos-Renyi, and Preferential Attachment).

The results, shown in Figure 4.5 (right), are quite reassuring: $L = 10$ seems to suffice across all the settings shown, and these results are also consistent with those obtained for Cycle and PA(2,1) networks. From this point on, results based on a MILP formulation use $L = 10$, unless otherwise specified.

4.7.3 Comparison of the Alternative Formulations

We offered several alternative formulations of the defender's optimization problem: MINLP (the mixed integer non-linear programming approach in which we explicitly encode attacker target choices), NLP (non-linear program in which attacker target choices are implicit), and two MILPs, the first that does encode target choices, which we call "MILP (baseline)", and the second that does not, and which we refer to as "MILP(reduced)".

We compare all these formulations in terms of objective value (i.e. average v_0 over 100 random realizations of target values and network topologies) and average running time. The results in Figure 4.8 (left) suggest that there is not a significant difference in efficacy of the programming approaches we propose. Running time, however, does in fact differentiate them. Experimentally we found that MINLP running time diverges rapidly from that of MILP: even with as few as 9 targets, KNITRO solver takes nearly 300 seconds, as compared to under 2 seconds solving the corresponding MILP approximation using CPLEX. Surprisingly, we found little difference in running time between the two MILP formulations, but the difference between MILP and NLP formulations is rather dramatic. Figure 4.8 (right) shows that the NLP formulation scales considerably better than MILP, solving instances with as many as 1000 targets in under 200 seconds (MILP already begins to reach its limit by $n = 50$). Interestingly, graph topology seems to play some role in determining the difficulty of the problem: Cycle graphs are solved much faster by NLP than Erdos-Renyi analogs.

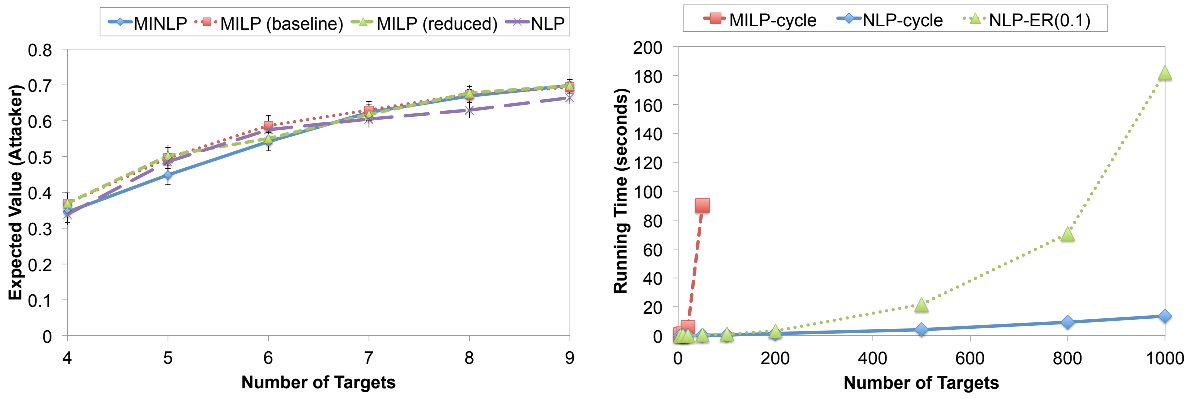


Figure 4.8. Left: Comparison of average attacker utility achieved using MINLP, two versions of MILP, and NLP formulations, using the Cycle topology. Right: Running time comparison between MILP and NLP on Cycle and ER(0.1) graphs. We omit MINLP which does not scale, and the two MILP formulations yield similar results, so we only present MILP (baseline) here.

4.7.4 Attacks Taking Multiple Steps

Approximation and Runtime Tradeoff

As our formulation of the defender's optimization problem in the case when attacks can take more than a single step to unfold allows one to make a principled tradeoff between runtime and approximation quality, we now study this tradeoff. Specifically, we fix the number of steps an attack takes at $h = 3$, fix the number of targets at 10, and vary $1 \leq K \leq 3$,.

The results are shown in Table 4.2. It is quite clear that solving this problem optimally is an unlikely proposition: even with $h = 3$ and only 10 targets, solving to optimality requires, on average, over 20 minutes. Fortunately, it appears that both $K = 1$ and $K = 2$ approximations achieve near-optimal utility, and are much faster.

K	expected utility	runtime (s)
1	0.52±0.01	0.3±0.02
2	0.48±0.01	7.18±1.16
3	0.47±0.01	1325±243

Table 4.2. Comparison of attacker's expected value and defender's network design cost for the NLP (ND) formulation solved by IPOPT and KNITRO, and the MILP (ND) formulation. For all, the number of targets is 20 and per-edge cost is 0.02. For KNITRO, we used 4 restarts; we had not tried more, as even with 4 a significant fraction of instances (between 5 and 10%) simply stall.

The Impact of Increasing the Length of Attack

Having observed that the defender does not lose very much by considering $K = 1$, we use this approximation to study the impact of increasing the length of an attack on attacker's expected utility. Specifically, the results in Figure 4.9 show the expected utility of the attacker, as well as the average running time, as a function of the number of time steps an attack takes to unfold. As we would expect, increasing the number of time steps decreases attacker utility, and increases running time; this is true for several values of the discount factor δ. Perhaps surprisingly, however, discount factor does not have an impact on attacker utility here, except when $h = 1$. A very high discount factor does, however, result in a higher running time: as the attacker becomes more patient, the computation must be increasingly subtle to ensure a highly efficacious patrolling policy. Nevertheless, we can ultimately observe that the running time scales extremely well with h when $K = 1$: even when $h = 10$, computing the defender policy is still on the order of 1 second.

4.7.5 Experiments with Discount Factor

Here we study the impact of changing the discount factor δ on the attacker's expected utility and the runtime of the NLP model. Figure 4.10 (left) shows that once the discount factor is at 0.5 or lower, it does not pay for the attacker to wait, and the utility is therefore insensitive to changing the discount factor in this region (recall that positive utility is attained only upon a successful attack in this setup, so attacking immediately implies that the discount factor plays no role, except to further discourage waiting). Considering the upper range of discount factors, we can observe that when $\delta > 0.75$, the attacker can often gain a non-negligible value from waiting, and, on the other hand, the expected utility at $\delta = 0.95$ is still significantly below that for $\delta = 1$, suggesting that qualitative differences exist between the two regimes.

Inspecting the runtime plot (Figure 4.10, right) reveals no significant runtime differences as long as the discount factor is below 0.95, but runtime rises sharply when it is higher.

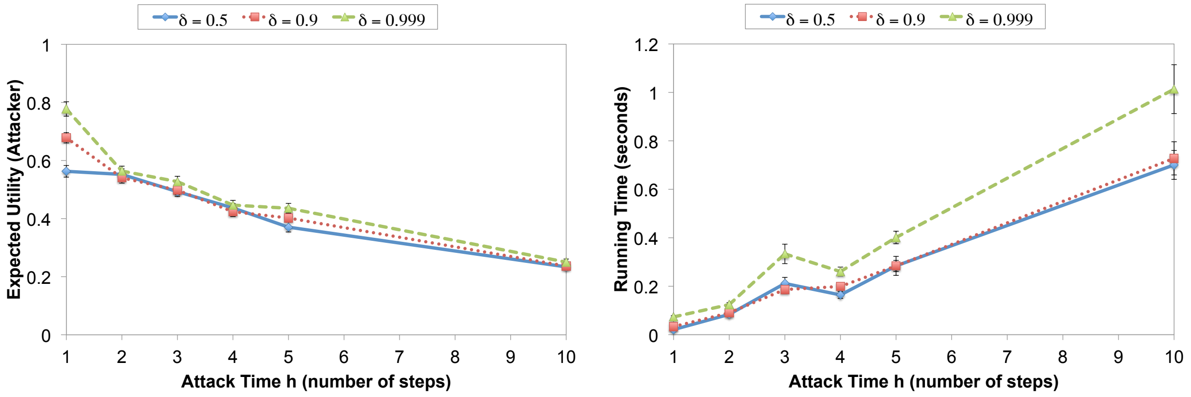


Figure 4.9. Attacker utility (left) and running time (right) as a function of the number of time steps the attacker takes to attack. The number of targets is fixed at 10, and graphs are generated according to $ER(0.1)$, with the base of operations (target 0) connected to all others as before. Problem are solved using IPOPT.

4.8 Adversarial Patrolling Games: Network Design

Thus far we assumed that the network constraining defender moves is given exogenously. A natural question is: what if the defender can build this network? For example, in a border patrol setting, the defender may choose to build roads or clear certain areas to enable direct moves between important checkpoints. Such investments to improve patrolling efficacy will usually be costly (particularly if one includes maintenance costs), but may be well worth the investment if targets are important enough.

Formally, suppose that the defender will first decide which edges to construct, with a directed edge from i to j costing c_{ij}. (Observe that we can allow for existing edges by setting the corresponding costs $c_{ij} = 0$, and can incorporate constraints by letting $c_{ij} = \infty$.) Once the graph is constructed, the adversarial patrolling game commences just as described above, and, thus, in making the decisions about which edges to construct, the defender must account for the impact of the resulting graph on patrolling efficacy. Fortunately, the decision to build edges can be incorporated directly into the mathematical programming formulations above, with A_{ij} now becoming variables, rather than specified problem parameters.[4]

[4] There is a subtle issue in the network design problem: the result that we rely on to allow us to consider only stationary Markov policies for the defender assumes a zero-sum game, which this no longer is. However, the setting is a zero-sum game *once the edges have been formed*, and that is all that we actually require.

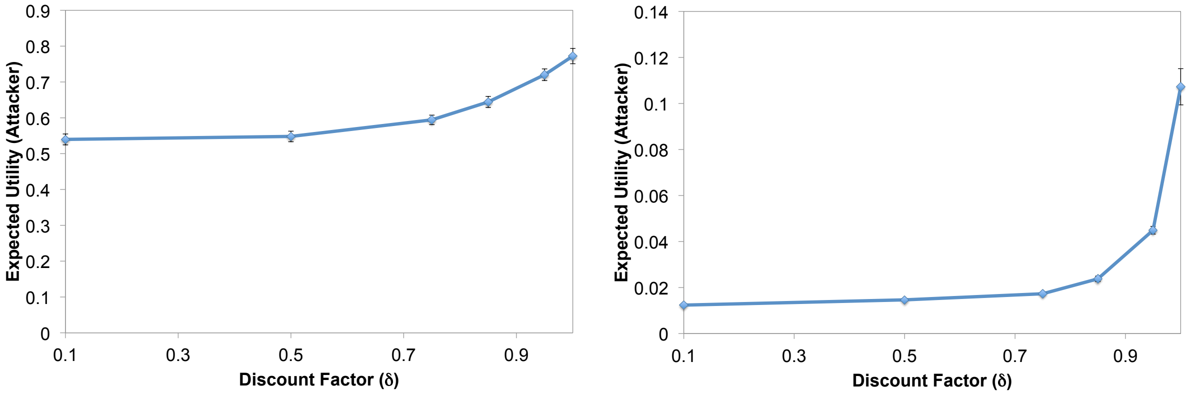


Figure 4.10. Left: Results for objective value of attacker in the baseline model as we vary the discount factor δ between 0.1 (very impatient attacker) and 1 (no discounting). The NLP model (solved with IPOPT) is used throughout, and the number of targets is fixed at 10. Right: Runtime of the baseline NLP model (solved with IPOPT) as we vary the discount factor δ between 0.1 (very impatient attacker) and 1 (no discounting). The number of targets is fixed at 10.

4.8.1 Baseline Network Design Formulation

One way to solve the network design problem would be to search exhaustively through all the networks: create a network, solve for defender utility using the approach from Section 4.6, and iterate. Intuitively, what we do here is short-circuit this approach by doing the entire optimization in one shot.

Let A_{ij} be binary variables with $A_{ij} = 1$ if and only if the defender builds an edge from i to j which he can subsequently use in patrolling decisions. The lone term involving A_{ij} in all our formulations above is linear in A_{ij}, and we therefore need to make no further modifications to the constraints. Since edges have a cost, we must change the objective to reflect the resulting cost-benefit tradeoffs. Therein lies a problem: our formulations above used $\sum_i v_i$ as an objective, while the defender's concern is only about v_0. Consequently, if we simply add a total incurred cost to $\sum_i v_i$ in the objective, the cost term will not be given sufficient weight, and the solution may be suboptimal: in fact, it is fundamentally the tradeoff between value and cost of building edges that we are trying to make here. The true objective of $v_0 + cost$, however, does not work either, since it will fail to correctly compute the values v_i of all states i, which are necessary to correctly obtain v_0: coefficients on all v_i must be strictly positive. We therefore offer the following approximate objective function:

$$\min \quad (1-\alpha)v_0 + \alpha \sum_{i \neq 0} v_i + \sum_{i,j} c_{ij} A_{ij},$$

where $\alpha > 0$ is some small real number, and the last term computes the total cost of building the graph. We now show that α can be scaled low enough to ensure that the resulting solution is arbitrarily close to optimal.

First, let us abstract the constraint set of the above optimization problem as some set $\mathscr{C}$. Let

$$(\pi^*, v^*, A^*) \in \arg \min_{(\pi,v,A)\in\mathscr{C}} \left(v_0 + \sum_{i,j\in T} A_{ij}c_{ij} \right)$$

be a true minimal (optimal) solution, with u^* the corresponding optimal utility, while

$$(\hat{\pi}, \hat{v}, \hat{A}) \in \arg \min_{(\pi,v,A)\in\mathscr{C}} \left((1-\alpha)v_0 + \alpha \sum_{j\neq 0} v_j + \sum_{i,j\in T} A_{ij}c_{ij} \right),$$

with $\hat{u}$ the corresponding expected *actual* utility of the attacker at an approximate solution.

Proposition 4.8.1. *Suppose that* $0 \leq v_i \leq \bar{V}$ *for all targets* $i \in I$*. Then* $\hat{u} \leq u^* + \alpha n \bar{V}$*.*

Proof. Proof.

$$\begin{aligned} \hat{u} = \hat{v}_0 + \sum_{i,j\in T} \hat{A}_{ij}c_{ij} &= (1-\alpha)\hat{v}_0 + \alpha \sum_{j\neq 0} \hat{v}_j + \sum_{i,j\in T} \hat{A}_{ij}c_{ij} + \alpha(\hat{v}_0 - \sum_{j\neq 0} \hat{v}_j) \\ &\leq (1-\alpha)v_0^* + \alpha \sum_{j\neq 0} v_j^* + \sum_{i,j\in T} A_{ij}^*c_{ij} + \alpha(\hat{v}_0 - \sum_{j\neq 0} \hat{v}_j) \\ &\leq v_0^* + \sum_{i,j\in T} A_{ij}^*c_{ij} + \alpha(\hat{v}_0 + \sum_{j\neq 0} v_j^*) \leq u^* + \alpha n \bar{V}. \end{aligned}$$

□

In our setting the attacker receives a reward only once, when he actually attacks a target; consequently, $v_i \leq \max_j \max\{U_a^c(j), U_a^u(j)\}$ for all targets $i \in T$. Thus, $\bar{V} = \max_j \max\{U_a^c(j), U_a^u(j)\}$. If we further let $\max\{U_a^c(j), U_a^u(j)\} \leq 1$ for all targets j (this is true in all our experiments below), $\bar{V} = 1$, and our approximation incurs an additive error of at most $n\alpha$.

The modifications above can be made directly to both the NLP and MILP formulations of the adversarial patrolling problem. However, the modification introduces integer variables, which are especially problematic when with start with a non-linear program. Below we offer an alternative network design formulation in which no integer variables are present.

4.8.2 NLP Network Design Formulation

Above, we used the graph constraint from the basic APG formulations unchanged, and merely introduced A_{ij} as integer variables. Alternatively, we can modify the graph constraint to recover an equivalent formulation of the network design problem that contains no integer variables.

Consider the set of constraints

$$\pi_{ij}(1 - A_{ij}) = 0 \quad \forall\, i, j \in I \tag{4.8.1}$$

which are equivalent to those in Constraint 4.6.2d (when $A_{ij} = 0$, π_{ij} are forced to be 0). While we have just replaced linear constraints with those that are non-linear, the win comes from the fact that we can now relax A_{ij} to be real-valued.

Proposition 4.8.2. *Suppose that $A_{ij} \geq 0$ is unrestricted and $c_{ij} > 0$. Further, suppose that we replace the linear graph Constraint 4.6.2d in the network design formulation with Constraint 4.8.1. Then an optimal solution A_{ij} is binary-valued.*

Proof. Proof. Suppose $\pi_{ij} > 0$. The only way for the constraint to equal zero in this case is to force $A_{ij} = 1$. Alternatively, suppose that $\pi_{ij} = 0$. Then the value of A_{ij} is unrestricted. However, since $A_{ij} \geq 0$, any positive value of A_{ij} would carry a cost, and have no benefit to the objective value, since $\pi_{ij} = 0$ and this link is effectively unused. Therefore in an optimal solution, $A_{ij} = 0$. □

We note that we can make an analogous modification to the MILP network design formulation, but must subsequently linearize the new set of graph constraints. Nevertheless, we can prove that the resulting linearized version always results in binary-valued A_{ij} (details are in the appendix).

4.8.3 Experiments: Network Design

In this section, we compare the MILP formulation for network design, which we refer to as MILP (ND), and the non-linear programming formulation in Section 4.8.2, which we refer to as NLP (ND).

The results in Table 4.3 offer a compelling case for the MILP network design formulation: attacker values achieved are not very different, but NLP-based approaches are clearly quite suboptimal in terms of design costs, building far more edges than optimal.

method	attacker value	design cost
MILP (ND) (CPLEX)	0.82±0.014	0.45±0.0058
NLP (ND) (IPOPT)	0.78±0.044	7.35±0.29
NLP (ND) (KNITRO)	0.77±0.021	3.14±0.084

Table 4.3. Comparison of attacker's expected value and defender's network design cost for the NLP (ND) formulation solved by IPOPT and KNITRO, and the MILP (ND) formulation. For all, the number of targets is 20 and per-edge cost is 0.02. For KNITRO, we used 4 restarts; we had not tried more, as even with 4 a significant fraction of instances (between 5 and 10%) simply stall.

In the next set of experiments, we let the cost c_{ij} for every edge be a fixed value c, which we vary between 0 and 0.1. Figure 4.11 shows the attacker expected utility and algorithm runtime for

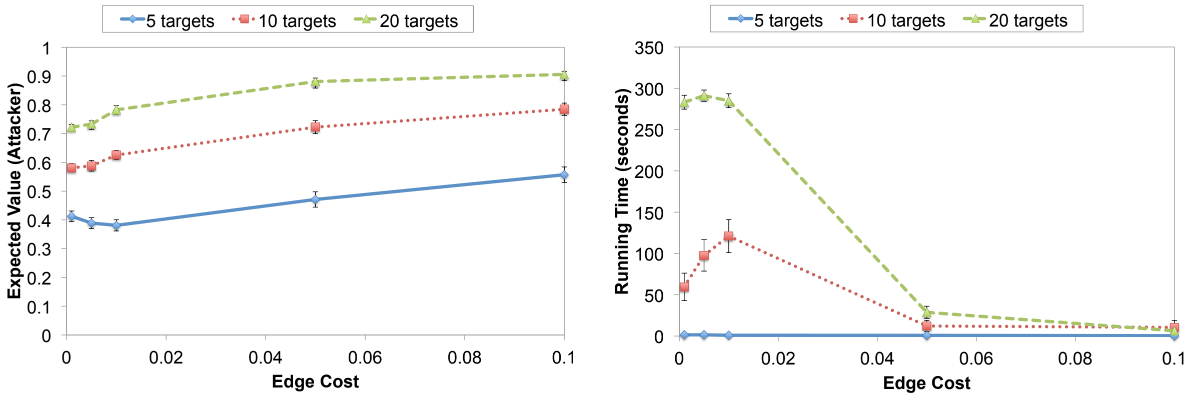


Figure 4.11. Network design: objective value and runtime for different edge costs and numbers of targets. Results from solving the MILP (ND) formulation (capped at 300 seconds).

varying costs per edge c and number of targets. Interestingly, at costs as low as 0.005, the expected utility is already nearly optimal (that is, we do essentially as well as when $c = 0$). For cost between 0.005 and 0.01, we see the peak in computational burden: edge costs are now non-negligible, but good solutions can still be obtained if only the most important edges are built.

4.9 Transition Costs

4.9.1 Formulation

In many realistic settings, rather than having a fixed graph that constrains defender's moves, we may posit that each directed edge (i, j) has some associated cost c_{ij} for the patroller to traverse, and the defender must decide at each point in time the most cost-effective way to patrol among all targets, depending on which target he is patrolling at the moment. (Notice that this setting is again a departure from our zero-sum assumption. In the sequel, we assume that stationary Markovian strategies nevertheless still suffice.) As an example, consider a border patrol setting: only a subset of targets is connected via easily traversable paths (e.g., roads), and in principle moves between targets separated by unfavorable terrain are not impossible, just substantially more costly. Depending on target value, patrol may at times wish to avail themselves of the more costly alternative routes.

Without loss of generality, suppose that the network is completely connected and remove the network constraint (Constraint 4.6.2d) from the optimization. Note that this is without loss of generality because for any edge with $A_{ij} = 0$ we can set the cost $c_{ij} = \infty$. Since the game is no longer zero-sum, the NLP formulations we have used cannot be easily extended to compute a Stackelberg equilibrium in this setting, as we need to introduce integer variables that explicitly represent attacker decisions. As such, we first extend the MINLP formulation to compute Markov stationary Stackelberg equilibria in general-sum APGs (Mathematical Program 4.6.1), and then note that we can convert the resulting formulation into a MILP approximation as we had done earlier. First, note that the objective function becomes $\min v_0 + C_0$, where C_0 computes total expected costs when the defender starts in state 0. Next, recall that b_i identifies whether the attacker attacks or waits in each state i, and we use it to compute total discounted cost C_i starting at each state i:

$$C_i = (1-b_i)\left(\sum_j \pi_{ij}c_{ij}\right) + b_i\left(\sum_j \pi_{ij}c_{ij} + \delta \sum_j \pi_{ij}C_j\right) = \sum_j \pi_{ij}c_{ij} + \delta \sum_j b_i \pi_{ij} C_j.$$

In the MILP approximation, we would additionally replace the variables π_{ij} with their discrete counterparts, obtaining

$$C_i = \sum_j \sum_l p_l d_{ijl} c_{ij} + \delta \sum_j \sum_l p_l b_i d_{ijl} C_j,$$

and then linearlize the non-linear constraint $b_i d_{ijl} C_j$ using McCormick inequalities, letting $h_{ijl} = b_i d_{ijl} C_j$, and ensuring that h_{ijl} satisfies the following set of constraints:

$$-Zb_i \leq h_{ijl} \leq Zb_i \;\forall\, i,j,l \tag{4.9.1a}$$

$$-Zd_{ijl} \leq h_{ijl} \leq Zd_{ijl} \;\forall\, i,j,l \tag{4.9.1b}$$

$$C_j - Z(2-d_{ijl}-b_i) \leq h_{ijl} \leq C_j + Z(2-d_{ijl}-b_i) \;\forall\, i,j,l. \tag{4.9.1c}$$

4.9.2 Experiments

In our experiments pertaining to the formulation that uses transition costs instead of a fixed graph, we generate the cost for each edge (i,j) i.i.d. from a uniform distribution on an interval $[0,c]$, where c is a parameter that we vary (we call it *cost upper bound*). The single exception is that we set the cost of staying at a given target to be 0, which seems natural in most realistic settings.

Figure 4.12 shows the attacker utility as well as total defender expenditures for 5 targets.[5] As expected, attacker value increases with defense costs, but rather gradually. Interestingly, the total costs of defense start gradually falling after reaching a peak around $c = 0.75$, presumably as some of the costs become so high so that the corresponding arcs are not worth taking no matter what target the value is.

4.10 Conclusion

We defined general-sum discounted stochastic Stackelberg games (SSG), presented a model of discounted adversarial patrolling on exogenous networks, and demonstrated how to formalize it as a highly structured SSG. We show that in general SSGs do not have Markov stationary Strong

[5] We used a time limit of 300 seconds for CPLEX to solve these problems. Doubling the time limit does not appreciably change the results.

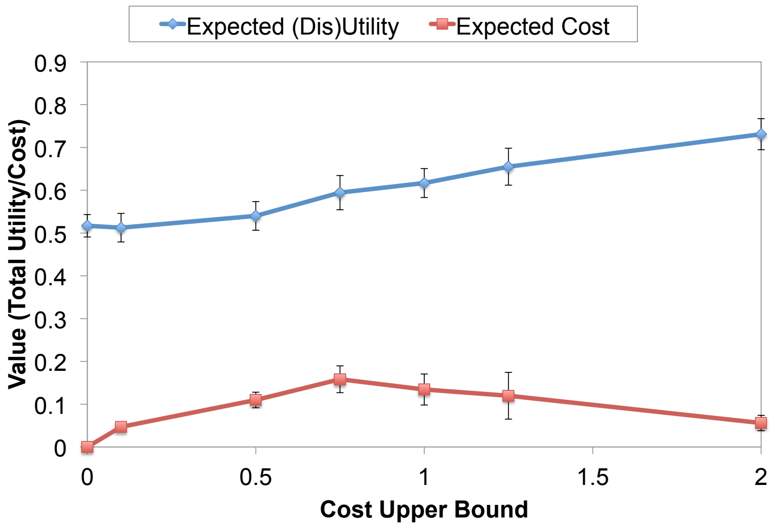


Figure 4.12. Attacker value v_0 and defender total expenditures, as a function of cost upper bound c in the "transition costs" model. Solved using the MILP with 6 discrete probability levels, for 5 targets. All results are based on at least 60 samples.

Stackelberg equilibria, even when they are of the restricted adversarial patrolling variety. However, we showed that team games do, indeed, have deterministic Markov stationary SSE. We then presented mathematical programming formulations for computing and approximating an SSE when the leader is restricted to Markov stationary policies, and showed that our approximation approach is convergent. We then adapted a known non-linear programming formulation to compute SSE in zero-sum adversarial patrolling games in two ways: the first introduced integer variables to compute the optimal attack utilities, following an approach commonly taken in the literature on Stackelberg games, while the second incorporated this decision directly into the NLP. Furthermore, we offered an alternative, albeit approximate, MILP formulation for this problem. We also presented two extensions of the baseline model and the corresponding NLP formulation: the first allows multiple defender resources, while the second incorporates multi-step attacks. Subsequently, we extended the baseline adversarial patrolling model to allow the defender to construct the graph constraining patrolling moves, at some per-edge cost, and offered NLP and MILP formulations to solve this problem. Finally, we presented a model in which the defender can move between an arbitrary pair of targets, but incurs a cost for each move, and offered NLP and MILP formulations to solve several variants of this problem.

Our experiments verify that solutions which we compute are significantly better than those obtained using an alternative formulation applied to a special case of *undiscounted* zero-sum APGs. Overall, both NLP and MILP formulations compute solutions much faster than mixed-integer nonlinear programs, while NLP is much faster than MILP, where applicable. On the other hand, we found that MILP computed much better solutions than NLP in the network design problem. Additionally, the "transition costs" model in which the defender is concerned with realized (rather than worst-case) costs does not lend itself to an easy NLP adaptation. Instead, we extended the MILP formulation which explicitly represents attacker target choices to compute approximate solutions in this case.

Chapter 5

Noncooperatively Optimized Tolerance: Decentralized Strategic Optimization in Complex Systems

5.1 Introduction

Highly optimized tolerance (HOT) and self-organized criticality (SOC) have received considerable attention as alternative explanations of emergent power-law cascade distributions [17, 26]. The SOC model [17, 37, 59] posits that systems can naturally arrive at criticality and power-law cascades, independently of initial conditions, by following simple rule-based processes. Among the important features of SOC are (a) self-similarity and homogeneity of the landscape, (b) fractal structure of cascades, (c) a small power-law exponent (i.e., heavier tails), and (d) low density and low yield (e.g., in the context of the forest fire model, described below). HOT [26, 27, 28, 85], in contrast, models complex systems that emerge as a result of optimization in the face of persistent threats. While SOC is motivated by largely mechanical processes, the motivation for HOT comes from evolutionary processes and deliberately engineered systems, such as the electric power grid. The key features of HOT are (a) a highly structured, self-dissimilar landscape, (b) a high power-law exponent, and (c) high density and high yield [28].

HOT and SOC can be cleanly contrasted in the context of the forest fire model [26, 59], which features a grid, usually two-dimensional, with each cell being a potential site for a tree. Intermittently, lightning strikes one of the cells according to some probability distribution. If there is a tree in the cell, it is set to burn. At that point, a cascade begins: fires spread recursively from cells that are burning to neighboring cells that contain trees, engulfing the entire connected component in which they begin (in our implementation, fires wrap around the grid walls, so there are effectively no boundaries). In the classical forest fire model (SOC) a tree sprouts in every empty cell with some fixed probability p. In contrast, the HOT model conceives of a global optimizer choosing the configuration of each cell (i.e., whether a tree will grow or not); what emerges globally as a consequence is a collection of large connected components of trees separated by “barriers” of no trees. The HOT model is deliberately robust to lightning strikes with the specified distribution; however, it is also extremely fragile to changes in the lightning distribution, whereas SOC does not exhibit such fragility. The HOT landscape tends to have a highly non-uniform distribution of “fire breaks”, or areas where no trees are planted, whereas the SOC landscape is homogeneous.

A natural criticism of the HOT paradigm is that, in complex systems, it is difficult to conceive of a single designer that manages to optimally design such a system. As a partial response, much work demonstrates that HOT yields qualitatively similar results when heuristic optimization or an evolutionary process is used [27, 114]. Still, most complex systems are not merely difficult to design globally, but are actually *decentralized*, with many entities responsible for parts of the whole system. Each entity is generally not motivated by global concerns, responding instead to individual incentives. For example, the Internet is fundamentally a combination of autonomous entities making their own decisions about network topology, protocols, and composition.

Our central contribution is to model complex systems as complex patterns of strategic interactions among self-interested players making independent decisions. We conceive that out of *strategic interactions* of such self-interested players emerges a system that is optimized *jointly* by all players, rather than *globally* by a single "engineer". Thus, we call our model *noncooperatively optimized tolerance* (NOT). Formally, our model is game theoretic, and we seek to characterize emergent properties of the system in a Nash equilibrium.

5.2 A Game Theoretic Forest Fire Model

Suppose that each player controls a portion of a complex system and is responsible for engineering his "domain of influence" against perceived threats. The interests of different players may be opposed if, say, an action that is desirable for one has a negative impact on another. Such interdependencies (commonly referred to as *externalities*) form a central aspect of our model. However, HOT arises as a special case of our construction, when the game has a single player.

We begin by introducing some general game theoretic notions, and then instantiate them in the context of a forest fire model. A *game* is described by a set of players I, numbering $m = |I|$ in all, where each player $i \in I$ chooses actions from a strategy set S_i so as to maximize his *utility* $u_i(\cdot)$. Notably, each player's utility function depends on the actions of other players as well as his own, and so we denote by $u_i(s) = u_i(s_i, s_{-i})$ the utility to player i when he plays a strategy s_i and others jointly play $s_{-i} \equiv (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_m)$, where these combine to form a joint strategy profile $s = (s_1, \ldots, s_i, \ldots, s_m)$.

We implement the game theoretic conception of complex system engineering in the familiar two-dimensional forest fire model, thereby allowing direct contrast with the now mature literature on HOT and SOC. In the NOT forest fire model, each player is allotted a portion of the square grid over which he optimizes his yield less cost of planting trees. [1] Let G_i be the set of grid cells under player i's direct control, let s_i be player i's strategy expressed as a vector s_i in which $s_{i,g} = 1$ if i plants a tree in grid cell g and $s_{i,g} = 0$ otherwise, and let $\Pr\{ g = 1 \mid s, s_{i,g} = 1 \}$ be the probability (with respect to the lightning distribution) that a tree planted in cell g survives a fire given the joint strategy (planting) choices of all players. Denote by s the vector of all players' choices. Since

[1] We note the resemblance of our grid division into subplots to the framework studied by Kauffman et al. [64], which divides a lattice in a similar manner, but with the goal of studying joint optimization of a global objective, rather than strategic interactions among players controlling different plots and having different goals.

exactly one player controls each grid cell, we simplify notation and use $s_g = s_{i,g}$ where i is the player controlling grid cell g. Let $N_i = |G_i|$ be the number of grid cells under i's control and ρ_i be the density of trees planted by i,

$$\rho_i = \frac{1}{N_i} \sum_{g \in G_i} s_g.$$

Let

$$Y_i(s) = \sum_{g \in G_i} \Pr\{ g = 1 \mid s \} s_g$$

be the yield for player i (it is convenient to define the yield as an absolute number of trees). Let c denote the cost of planting a tree. The utility of player i is then

$$u_i(s) = \sum_{g \in G_i} (\Pr\{ g = 1 \mid s \} - c) s_{i,g} = Y_i(s) - cN_i\rho_i.$$

The result of joint decisions by all players is a grid that is partially filled by trees, with overall density $\rho(s)$ and overall yield $Y(s)$ given by a sum ranging over the entire grid G, i.e., $Y(s) = \sum_{g \in G} \Pr\{ g = 1 \mid s \} s_g$. Let N be the number of cells in the entire grid. We then define *global utility (welfare)* as

$$W(s) = \sum_{i \in I} u_i(s) = Y(s) - cN\rho(s).$$

Note that when $m = 1$, $W(s)$ coincides with the lone player's utility. A part of our endeavor below is to characterize $W(s^*)$ and $\rho(s^*)$ when s^* is a Nash equilibrium, defined as a configuration of joint decisions by all players such that no individual player can gain by choosing an alternative strategy (planting configuration) s'_i *keeping the decisions of other players fixed.*

We systematically vary several model parameters. The first is the number of players m, which we vary from $m = 1$ to N, fixing the size of the grid at $N = 128 \times 128$.[2]. The former extreme corresponds precisely to the HOT setting, while in the latter the players are entirely myopic in their decision problems, each concerned with only a single cell of the grid. The negative externalities of player decisions are clearly strongest in the latter case. The entire range of player variation is $m \in \{1, 2^2, 4^2, 8^2, 16^2, 32^2, 64^2, 128^2\}$. The second parameter that we vary is the cost of planting trees: $c \in \{0, 0.25, 0.5, 0.75, 0.9\}$. Finally, we vary the scale of the lightning distribution, which is always a truncated Gaussian centered at the top left corner of the grid. We let the variance (of the Gaussian before truncation) be N/v, and vary $v \in \{0.1, 1, 10, 100\}$. For example, at $v = 0.1$ the distribution of lightning strikes is approximately uniform over the grid, while at $v = 100$ the distribution is highly concentrated in the top left corner. We divide the grid among m players by partitioning it into m identical square subgrids, ensuring throughout that m is a power of 4.

[2] This was the largest grid size on which we could approximate equilibria in reasonable time.

5.3 Analysis of the NOT Forest Fire Model

5.3.1 Characterization of 1- and N-player Settings in the 1-D Case

We begin the analysis by considering the two extremes, $m = 1$ and $m = N$, in a simpler model where the forest fire grid is one-dimensional (i.e., a line) and the lightning distribution is uniform. This analysis will provide some initial findings and intuition that we then carry over into the more complex two-dimensional case.

Without loss of generality, let k be the length of a sequence of planted cells (1's) followed by l unplanted cells (0's) and suppose that $1 \ll k \ll N$.

First, consider the case with $m = 1$ and assume that $c < 1 - 1/N$. Assume that k is identical for all sequences of 1's (when $k \ll N$, this is almost with no loss of generality, since 1's can be swapped, keeping the density constant, without changing the utility) and note that in an optimal solution $l = 1$. The utility of the player (and global utility) is then

$$u_i(k) = W(k) = \sum_{g \in G} (\Pr_f\{g = 1 \mid s\} - c)s_g = N\rho(k)\left(1 - \frac{k}{N} - c\right),$$

where $\rho(k) = k/(k+1)$. This function is concave in k. To see this rewrite u_i as

$$u_i(k) = \frac{Nk(1-c) - k^2}{k+1}.$$

Taking the first derivative, we get

$$u_i' = \frac{N(1-c) - k^2 - 2k}{(k+1)^2}.$$

Differentiating again we get

$$u_i'' = -\frac{2(1 + N(1-c))}{(k+1)^3} < 0,$$

and, hence, u_i is concave in k.

Thus, treating k as a continuous variable, which is approximately correct when $k \gg 1$, the first-order condition gives us the necessary and sufficient condition for the optimal k^*. This condition is equivalent to

$$k^2 + 2k - N(1-c) = 0.$$

The solutions to this quadratic equation are

$$k = \frac{-2 \pm \sqrt{4 + 4N(1-c)}}{2}.$$

Since k must be positive, we can discard one of the solutions, leaving us with

$$k^* = \sqrt{N(1-c)+1} - 1.$$

Evaluating ρ and W at k^*, we get

$$\rho(k^*) = \frac{\sqrt{N(1-c)+1}-1}{\sqrt{N(1-c)+1}}$$

and

$$u_i(k^*) = W(k^*) = \rho(k^*)(N(1-c) - \sqrt{N(1-c)+1}-1).$$

We can observe that $\rho(k^*)$ tends to 1 as N grows, while $W(k^*)$ tends to $N(1-c)$.

Consider next the case with $m = N$. While there are many equilibria, we can precisely characterize upper and lower bounds on k and l, and, consequently, the set of equilibria. First, we note that l must be either 1 or 2; otherwise, by the assumption that $c < 1-1/N$, the player governing any grid cell that is not adjacent to a sequence of 1's will prefer to plant a tree. Formally, we first note that by definition, $l > 0$. Suppose $l > 2$ and, thus, there is a player not planting a tree who is not adjacent to another with $s_g = 1$. Then his utility from planting is $1-1/N-c$, and he (weakly) prefers not to plant as long as $1-1/N-c \leq 0$ or $c \geq 1-1/N$, which is ruled out by our assumption that $c < 1-1/N$.

Second, we can get an upper bound on k by considering the incentive of a player that is part of the sequence of 1's. This player will prefer to plant as long as $1-k/N-c \geq 0$, giving us $k^E \leq N(1-c)$. A well-known measure of the impact of equilibrium behavior on global utility is the "price of anarchy", the ratio of optimal global utility, here $W(k^*)$, to global utility at the worst-case equilibrium [71, 99, 86]. The upper bound on k^E gives us the worst-case equilibrium from the perspective of global utility, with $W(k^E) = 0$ resulting in an infinite price of anarchy (that is, global utility in the worst-case equilibrium is arbitrarily worse than optimal for a large enough number of players and grid cells N).

Looking now at the lower bound on k^E, we can distinguish two cases, $l = 1$ and $l = 2$. When $l = 2$, either player not planting a tree prefers not to plant as long as $1-(k+1)/N-c \leq 0$, and, therefore, $k^E \geq N(1-c)-1$. For $l = 1$, suppose that the two sequences of 1's on either side of the non-planting player have lengths k and k'. The player will prefer not to plant as long as $1-(k+k'+1)/N-c \leq 0$, where we are adding k and k' since he will be joining the two sequences together if he plants. This gives us $k+k' \geq N(1-c)-1$. Since we are after a lower bound, suppose without loss of generality that $k \leq k'$. We then get $k^E \geq [N(1-c)-1]/2$. It is instructive to apply now another measure of the impact of equilibrium behavior, the "price of stability", defined as the ratio of optimal global utility to global utility at the *best-case* equilibrium [86, 14]. The best-case equilibrium in our case has $l = 1$ and $k^E = [N(1-c)-1]/2$, and the asymptotic price of stability is 2.

Now we compare the density at equilibrium and at the optimal configuration. We are looking for the conditions under which the equilibrium density is strictly higher. Notice that it certainly isn't always the case. For example, if $c > 1-1/N$, no trees will be planted at all in equilibrium or in an optimal configuration. Consequently, the density will be 0 in both cases. When $N(1-c) \gg 1$ and N is large, the density in the best-case equilibrium is

$$\rho(k^E) = \frac{N(1-c)-1}{N(1-c)+1}.$$

Thus, $\rho(k^E) > \rho(k^*)$ iff

$$\begin{aligned}
\frac{N(1-c)-1}{N(1-c)+1} &> \frac{\sqrt{N(1-c)+1}-1}{\sqrt{N(1-c)+1}} \\
\Leftrightarrow (N(1-c)-1)\sqrt{N(1-c)+1} &> (N(1-c)+1)(\sqrt{N(1-c)+1}-1) \\
\Leftrightarrow 2\sqrt{N(1-c)+1} &< N(1-c)+1 \\
\Leftrightarrow 4(N(1-c)+1) &< (N(1-c))^2+2N(1-c)+1 \\
\Leftrightarrow 2N(1-c)+3 &< (N(1-c))^2.
\end{aligned}$$

Solving the corresponding quadratic inequality gives us the condition that

$$N(1-c) > 3.$$

Since we assume $N(1-c) \gg 1$ throughout, we effectively have that $\rho(k^E) > \rho(k^*)$ under the assumptions operational here.

5.3.2 Equilibria When $c = 0$ and $m = N$

Next, consider a special case in the 2-D forest fire model when $c = 0$ and $m = N$ (i.e., when each player controls a single grid cell). In this case, there are only two pure strategy Nash equilibria: one with every player planting a tree, and another with a single player not planting. Indeed, planting is a weakly dominant strategy for every player. To see this, suppose that the number of players planting is $z < N-1$, and consider a player who is not planting a tree. If he decides to plant, the probability of his tree burning down is at most $(N-1)/N < 1$, and so the player has a strict incentive to plant. Furthermore, since there is no cost of planting, any player who is planting a tree does not lose anything by doing so. Thus, every player strictly prefers to plant as long as $z < N-1$, and weakly prefers to plant when $z = N-1$ (in which case expected utility is zero whether he plants or not). Finally, every player planting is clearly an equilibrium, and the only other equilibrium has a single player who does not plant (since he is indifferent, and every other player strictly prefers to plant if that player does not).

5.3.3 Computational Analysis of the 2-D Forest Fire Model

Equilibrium Approximation

A full analysis of the two-dimensional model in all the relevant parameters is beyond mathematical tractability. Furthermore, the problem of computing exact equilibria, or even exact *optima* for any player, is intractable, as the size of the space of joint player strategies in our setting is 2^{16384}. Nevertheless, it turns out that simple iterative algorithms for approximating equilibria as well as optimal decisions by individual players are extremely effective. Specifically, we use a variant of *best response dynamics* for approximating Nash equilibria, which iteratively optimizes each player's strategy, keeping strategies of other players fixed [51]. (We found that both asynchronous and partially synchronous versions of best response dynamics yield similar results; below we report on the asynchronous implementation.) Within this procedure, we approximate optimal responses of individual players using *sampled fictitious play* [104]. In sampled fictitious play, each grid cell controlled by player i becomes a "player" in a cooperative subgame (where each cell has i's utility as its goal), and random subsets of cells are iteratively chosen to make simultaneous optimizing decisions.

We now present the details of the algorithms we used to approximate equilibria. First, we show the "outer loop" algorithm for best response dynamics as Algorithm 2. The parameter T_{br} varies depending on the number of players. For example, if there is just one player, $T_{br} = 1$, whereas $T_{br} = 50$ when $m = N$. The variation is a consequence of extensive experimentation looking at sensitivity of results to increasing the number of iterations. Our values are high enough that results do not change appreciably when the number of iterations increases. We set $p_{player} = 0.9$.

```
$s_g \leftarrow 0 \; \forall g \in G$
**for** $n = 1$ to $T_{br}$ **do**
  **for** $i = 1$ to $m$ **do**
    Fix $s_{-i}$
    **if** RAND $\leq p_{player}$ **then**
      $\hat{s}_i \leftarrow$ OPT$(s_{-i})$
    **else**
      $\hat{s}_i \leftarrow s_i$
    **end if**
    $s_i \leftarrow \hat{s}_i$
  **end for**
**end for**
```

Algorithm 2: BestResponseDynamics(T_{br}, p_{player})

For each player selected by the random biased coin flip ("RAND" is a uniform random number on the unit interval), the algorithm calls OPT() to approximate the best response of the player to a fixed grid configuration chosen by the others. Our choice for this procedure is sampled fictitious play, which is shown in pseudocode as Algorithm 3.

Here, RAND() when called with a list argument picks a uniformly random element of the list. $u_i()$ is a call to an oracle (a simulator) to determine i's utility in a particular grid configuration. $u_i(s_g = a, s'_i, s_{-i})$ denotes utility when i plays according to s'_i, except he sets $s_g = a$. We set history size $h = 1$ and exploration parameter $\alpha = 0$. Thus, each grid cell at iteration t is always best-responding to the grid configuration from iteration $t-1$. We set $p_{cell} = \max\{0.05, 1/N_i\}$. Thus, on average, one player best-responds in each iteration. Our parameters for both the optimization routine and the best response routine were chosen based on extensive experimentation. Specifically, we sought to increase the number of iterations until the point at which results no longer appreciably change. Similarly, the probability of choosing a player was increased until the results were relatively insensitive to further change. This is demonstrated for two of the parameters, num-

```
$s_g \leftarrow 0 \; \forall g \in G_i$
$H \leftarrow ()$    // Initialize history of past choices $H$ to an empty list
for $n = 1$ to $T_{opt}$ do
  $s'_i \leftarrow$ ChooseActions$(i, \alpha, H)$
  $\hat{s}_i \leftarrow s_i$
  for $g \in G_i$ do
    if RAND $\leq p_{cell}$ OR $|G_i| = 1$ then
      if $u_i(s_g = 1, s'_i, s_{-i}) > u_i(s_g = 0, s'_i, s_{-i})$ then
        $\hat{s}_g \leftarrow 1$
      else
        $\hat{s}_g \leftarrow 0$
      end if
    end if
  end for
  append_back$(H, \hat{s}_i)$    // Add $\hat{s}_i$ at the end of list $H$
  if $|H| > h$ then
    remove_front$(H)$    // Remove the first element
  end if
  if $u_i(\hat{s}_i, s_{-i}) > u_i(s_i, s_{-i})$ then
    $s_i \leftarrow \hat{s}_i$
  end if
end for
return $s_i$
```

Algorithm 3: OPT($s_{-i}, T_{opt}, p_{cell}, \alpha, h$)

ber of optimization iterations T_{opt} and the probability of choosing a player p_{player} in best response dynamics in Figure 5.1 (left and right plots respectively). In both cases, our specific parameter values of 200 and 0.9 respectively are conservative choices. The parameter α of the optimization routine was chosen to be 0 after it was observed that decreasing it always improved the result of optimization.

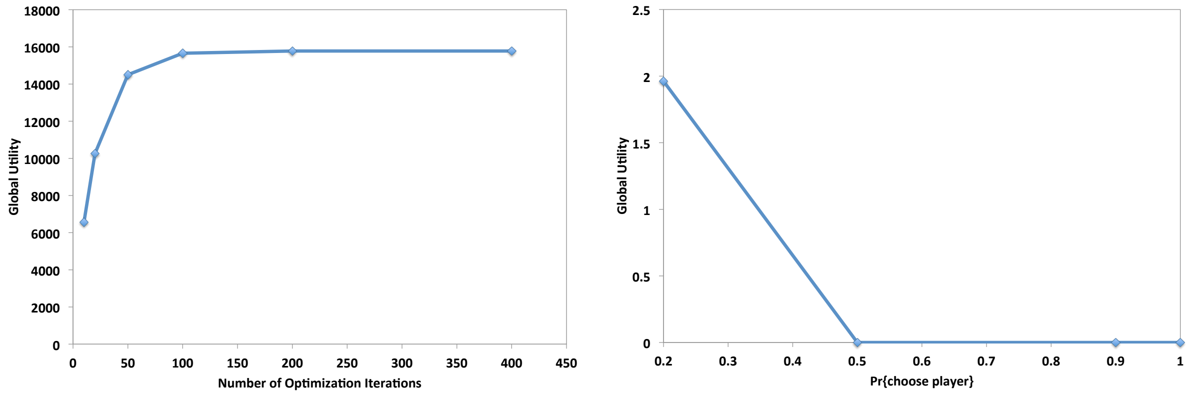


Figure 5.1. Examples of sensitivity analysis. Left: impact of increasing the number of optimization iterations when $m = 1$. Right: impact of increasing the probability of choosing a player in the best response dynamics routine when $m = N^2$. We set $c = 0$ in both.

Algorithm 3 uses the subroutine ChooseActions(), which is specified as Algorithm 4. In Table 5.1 we specify the number of iterations used for the outer loop (best response dynamics) and inner loop (approximate optimization).

```
for $g \in G_i$ do
  if RAND $\leq \alpha$ OR $H = ()$ then
    $s_g \leftarrow$ RAND$((0,1))$
  else
    $s_g \leftarrow$ RAND$(H)_g$
  end if
end for
return $s_i$
```

Algorithm 4: ChooseActions(i, α, H)

Global Utility

Our first question concerns the variation of global utility $W(s^*)$ with the number of players m, the cost c, and the parameter v governing variance of the lightning distribution. First, recall that $W(s^*)$

# players	T_{br}	T_{opt}
1	1	200
4	5	120
16	20	80
64	20	80
256	20	80
1024	40	80
4096	20	35
16384	50	1

Table 5.1. Numbers of iterations of best response dynamics and sampled fictitious play in the 2nd and 3rd column respectively.

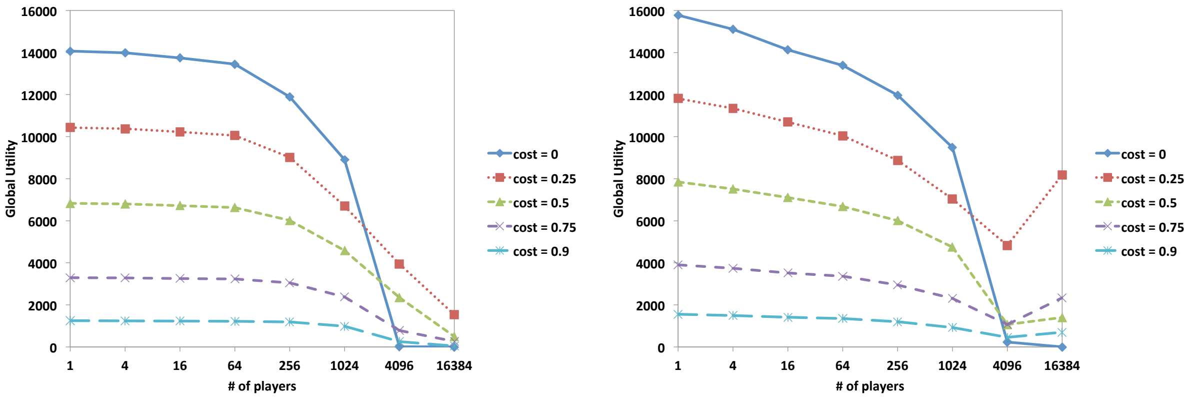


Figure 5.2. Global utility $W(s^*)$ as a function of m for $c \in \{0, 0.25, 0.5, 0.75, 0.9\}$. Left: $v = 0.1$ (nearly uniform distribution). Right: $v = 100$ (highly concentrated distribution).

will be no better than optimal for $m > 1$, and it seems intuitive that it is a non-increasing function of m. Additionally, we showed above that when $c = 0$ and $m = N$, we have a global utility of 0, since the only equilibria involve either all, or all but one, players planting trees. The question is: what happens in the intermediate cases? Figure 5.2 provides some answers. When $c = 0$, the initial drop in global utility is quite shallow for $m < 256$, particularly when the lightning distribution is relatively diffuse ($v < 100$). However, once the number of players is relatively large, global utility drops dramatically, and nearly reaches 0 already when $m = 4096$. For $c > 0$, the dropoff in global utility with the number of players becomes less dramatic.

Density and Fire Break Distribution

Our next task is to consider how the density changes with our parameters of interest. Based on the observation above, we expect the density to be 1, or nearly so, when $c = 0$ and $m = N$. The density should be appreciably below 1 when $m = 1$. Furthermore, the density should decrease with increasing cost c. In general, our intuition, based on all previous analysis, would suggest that density should increase with the number of players: after all, each player's decision to plant a tree does not account for the negative impact it has on other players. Working from this intuition, the simulation results in Figure 5.3 are highly counterintuitive: the overall density *falls* with increasing number of players until m reaches 1024, and only when the number of players is very high (4096 and N) is it generally higher than the optimal density. This dip is especially apparent for a highly concentrated lightning distribution ($v = 100$).

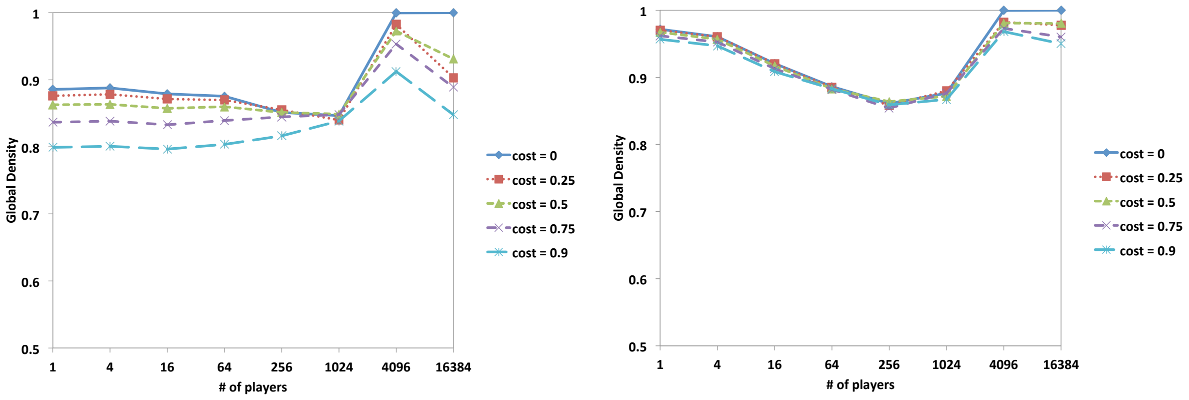
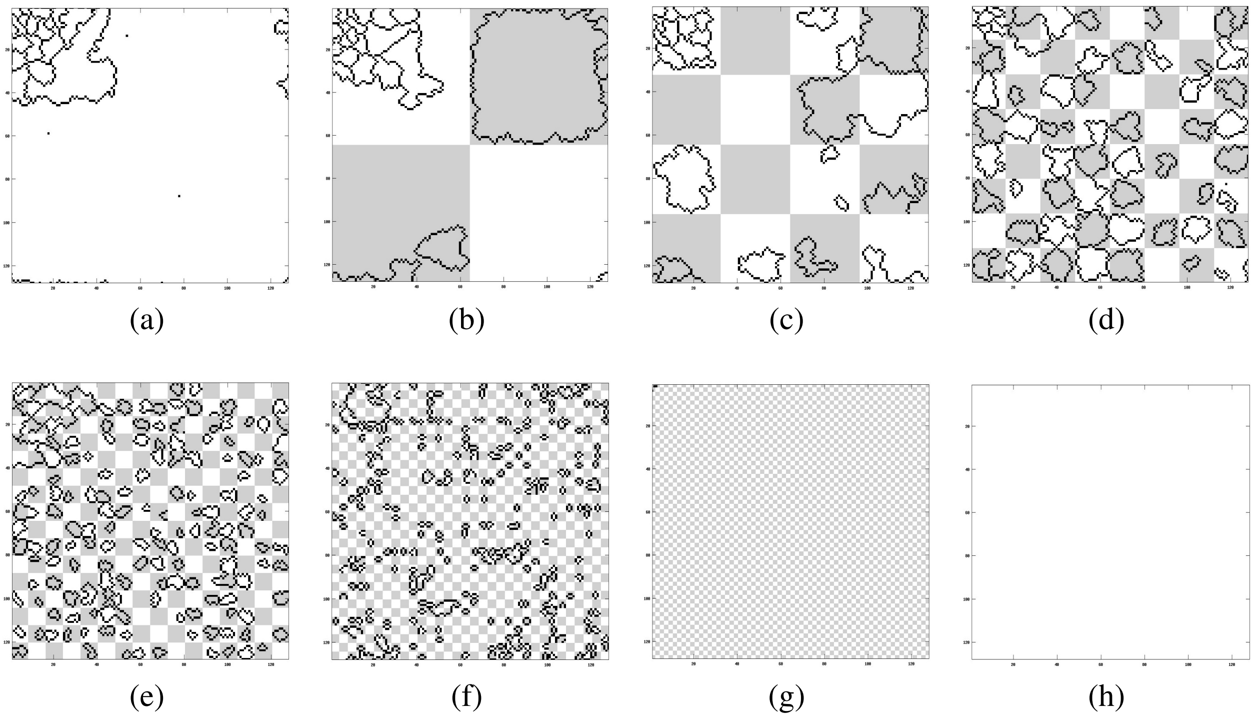


Figure 5.3. Density ρ as a function of m for $c \in \{0, 0.25, 0.5, 0.75, 0.9\}$. Left: $v = 0.1$ (nearly uniform distribution). Right: $v = 100$ (highly concentrated distribution).

To understand this phenomenon we refer to Figure 5.4, showing actual (approximate) equilibrium grid configurations for varying numbers of players when $c = 0$ and $v = 100$. We can observe that each player's myopic self-interest induces him to construct *fire breaks* in his territory where none exist in a globally superior single-player configuration. Thus, for example, contrast Figure 5.4



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 5.4. Sample equilibrium grid configurations with $c = 0$, $v = 100$, and the number of players varied between 1 and $N = 16384$. Blank cells are planted and marked cells are unplanted. Player domains of influence are shaded in a checkerboard pattern. (a) 1 player, equivalent to HOT; (b) 4 players; (c) 16 players; (d) 64 players; (e) 256 players; (f) 1024 players; (g) 4096 players; (h) 16384 players. To avoid clutter, we omit the checkerboard pattern with N players, where each grid cell contains a tree. Note that players adopt different strategies in similar conditions since best response is only approximate and stochastic, and there are likely many nearly optimal configurations.

(a) and (b). In the former, most of the grid is filled with trees, and much of the action happens in the upper left corner (the epicenter of the lightning distribution), which is filled with fire breaks that confine fires to relatively small fractions of the grid. In the latter, the upper left corner is now under the control of a single player, and other players find it beneficial to plant fire breaks of their own, since the "wasted" land amounts to only a small fraction of their landmass, and offers some protection against fire spread to the protected areas from "poorly" protected neighboring territories. With more players, we see coordination between neighbors emerge, as they jointly build mutually beneficial fire breaks, but such cooperation is not global, and becomes increasingly diffuse with greater number of players. Nevertheless, increasing the number of players results in a greater amount of total territory devoted to fire breaks by individual players or small local neighborhoods, and, as a result, an overall loss in planting density as observed.

Fragility

Since the density is decreasing for intermediate numbers of players, a natural hypothesis is that the fire breaks are distributed suboptimally. We can observe this visually in Figure 5.4.

Specifically, the equilibrium grid configurations suggest that the location of fire breaks becomes less related to the lightning distribution as the number of players grows. To measure this formally, we compute

$$C = \frac{\sum_{g \in G} p_g (1 - s_g)}{1 - \rho}.$$

The numerator is the probability that lightning strikes an empty (no tree) cell, where p_g is the probability of lightning hitting cell g, and s_g is the indicator that is 1 when g has a tree and 0 otherwise. The denominator is the fraction of the grid that is empty. The intuition behind this measure is that when fire breaks (i.e., empty cells) lie largely in regions with a high probability of lightning, C will be much larger than 1, whereas if empty cells are distributed uniformly on the grid, $E[C] = 1$ (these are formally shown in the next section) Figure 5.5 (left) confirms our hypothesis: initially, C is quite high, but as the number of players increases, C approaches 1. Interestingly, when the number of players is very large ($m = 4096$) this result reverses, with C jumping abruptly. To understand this phenomenon, note that when $m = 4096$, each player controls only a 2×2 subgrid, which is simply too small for a local fire break to be worthwhile unless the fire risk is very high. Thus, the only players with any incentive to build fire breaks are those close to the epicenter of lightning.

Considering the spatial distribution of empty grid cells apart from lightning strikes, we see in Figure 5.5 (right) that the centroid of the empty cells begins near the $(0,0)$ point, but approaches the center of the grid with increasing number of players.[3] Interestingly, even for a moderate number of players ($m = 16$), the distribution of fire breaks is nearly homogeneous and almost unrelated to the lightning distribution. This suggests that global utility would remain relatively robust to changes in the lightning distribution compared to the HOT model. To verify this, we show in Figure 5.6 aver-

[3] Here again we see that the center shifts back to near the $(0,0)$ point when $m = N/4$, for the same reasons we just outlined.

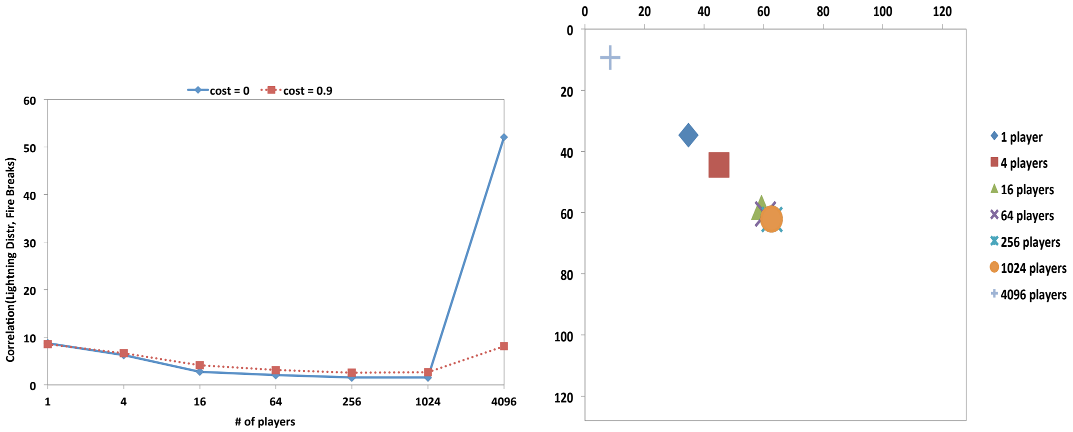


Figure 5.5. Left: a measure of correlation (C, defined in the text) between the lightning distribution and the fire breaks (empty cells) across subgrids for $c = 0$ and $c = 0.9$. As C approaches 1, the locations of empty cells become essentially unrelated to the distribution of lightning strikes. Right: centroid coordinates of the empty grid cells when $c = 0$ (the results are similar when $c = 0.9$).

age global utility of equilibrium configuration *after the lightning distribution is randomly changed*. Whether the cost of planting trees is high or low, the figure shows significantly reduced fragility for an intermediate number of players (between 16 and 1024). Indeed, when cost is high, the system remains less fragile than HOT even in the limiting case of $m = N$. Because global utility remains relatively close to optimal across a wide range of settings when m is below 256, our results suggest that the regime of intermediate numbers of players retains the robustness of HOT, while developing some features of SOC that make it less fragile to changes in the environment. Perhaps the most important reason for this phenomenon is the impact that negative externalities have on behavior of agents most susceptible to them: players closest to the epicenter of the lightning distribution tend to overplant, and others respond by building firebreaks around parts of their territory, partially protecting themselves from negative effects of neighbors' decisions. A direct consequence of these decisions is that the overall configuration remains quite robust to lightning strikes. A surprising consequence is that the resulting fire breaks form effective barriers preventing excessive spread of fire if the lightning distribution changes. When the number of players (m) is very small, however, player decisions correspond very closely to the actual lightning distribution, increasing fragility, while a very large m fragments decisions too much, and player decisions are highly myopic, with resulting configurations often not robust and highly fragile.

Distribution of Burnout Cascades

One of the central results of both SOC and HOT models is a power-law distribution of burnout cascades. Since our model generalizes HOT, we should certainly expect to find an approximately

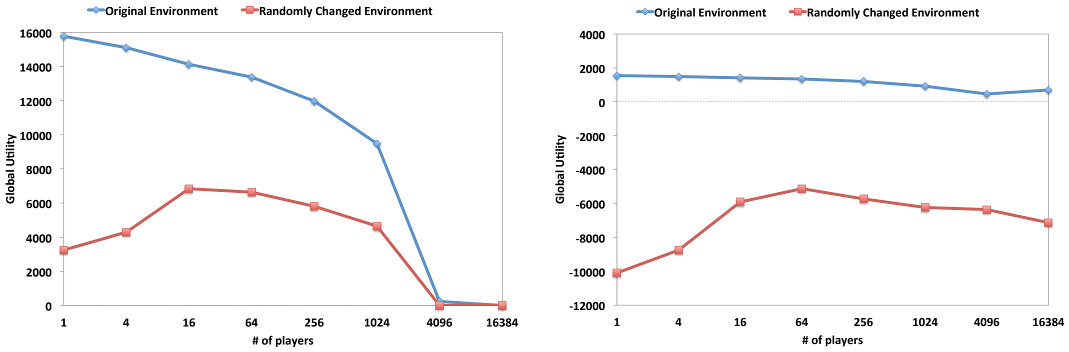


Figure 5.6. Fragility of NOT configurations for $v = 100$. Given the (approximate) equilibrium configurations generated for a lightning distribution centered at the upper left corner of the grid, we changed the lightning distribution by generating the center of the Gaussian uniformly randomly from all grid locations. We then evaluated expected global utility given the altered lightning distribution. The graph plots averages of repeating this process 30–80 times, as compared to global utility for the original environment. Left: $c = 0$. Right: $c = 0.9$.

power-law distribution in the corresponding special case of $m = 1$. We now study how the burnout distribution behaves with respect to the parameters of interest.

Figure 5.7 shows fire cascade distributions on the usual log-log plot for $v = 10$. When $m = 1$ (red points), the results suggest an approximate power-law distribution across a range of scales. Additionally, even when m is greater than 1 but relatively small (green points), the distribution remains approximately linear across a range of scales, suggesting that the power law is likely not unique to the HOT setting. Once the number of players is large, however, the distribution of cascades less resembles a power law, and begins to feature considerable curvature even at the intermediate scales. In that sense, the NOT setting with many players is unlike both HOT and SOC. The most important aspect of the cascade distributions is that the tails are systematically increasing with the number of players in all observed settings (this remains the case for Gaussians with greater and smaller variance, not shown here).

5.4 Discussion

The results described in the previous section show features of both HOT and SOC. When the number of players is small, the NOT setting closely resembles HOT, and, indeed, HOT is a special case when there is a single player. Perhaps surprisingly, features of HOT persist even when the number of players becomes larger, but as the number of players increases, we also begin to observe many features identified with SOC. The system retains its robustness to the lightning strikes—a key

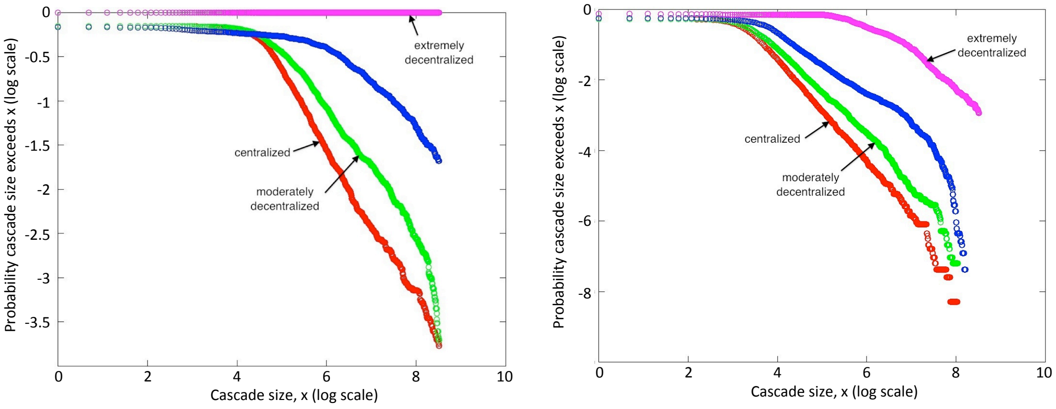


Figure 5.7. Distribution of tree burnout cascades, shown on a log-log plot with $\Pr\{X \geq x\}$ on the vertical axis and x on the horizontal axis, where X is the random variable representing cascade size. The plots feature (bottom to top) $m = 1$ (red), $m = 16$ (green), $m = 256$ (blue), and $m = 4096$ (purple), with the left plot corresponding to $c = 0$ and the right plot corresponding to $c = 0.9$. Both plots correspond to $v = 10$.

feature of HOT—even when the number of players is relatively large. It achieves this robustness in part due to the emergence of cooperation between neighboring players, who jointly build fire breaks spanning several players' territories. The cooperation required to retain near-optimal performance becomes increasingly difficult, however, as the system becomes highly fractured among small domains of influence.

As cooperation becomes less effective, players fall back on protecting their own domain of influence by surrounding it (or parts of it) with deforested land, so long as the fraction of land covered by trees is large enough to make this endeavor worthwhile. This gives rise to the counterintuitive result that the density of trees initially falls as the number of players increases.

Since even a moderately fractured landscape requires each player to focus on protecting his or her own domain, we observe decreasing correlation between locations of frequent lightning strikes and locations of fire breaks. With increasing number of players, this correlation systematically decreases, and the spatial distribution of empty cells becomes increasingly homogeneous—striking features of SOC that emerge even when the number of players is not very large and the global performance is still highly robust to lightning strikes. Thus, the intermediate range of players appears to exhibit both the robustness of HOT and the lack of fragility to changes in the lightning distribution associated with SOC.

Another feature of SOC in contrast to HOT is a heavier-tailed distribution of burnout cascades. We in fact observe that the tail of the burnout distribution becomes heavier with increasing number of players, superficially appearing to shift to an SOC regime. However, these distributions begin to substantially deviate from a power law even visually, and the setting is therefore in that respect entirely unlike the criticality observed in SOC.

Chapter 6

Decision and Game Theoretic Foundations of Risk and Trust

While much discussion has gone into evaluating risk and trust for a given system, more often than not these are not defined in terms of sound decision theoretic foundations. In this chapter, we attempt to provide some such foundations, and draw the connection to the common definitions actually used.

We begin by providing high-level, intuitive definitions for risk and trust, and then proceed to build a formal framework to think about these. We define *trust* in a given system to be the property that, roughly, *very bad things are unlikely to happen*. This informal definition has two crucial components upon which a framework can be based: *very bad things*, alluding to what is commonly know as *consequences*, and *unlikely to happen*, alluding to the probability that said things can happen to the system. We define *risk* of a system as the *expected losses* incurred, where the expectation is taken with respect to the events that result in such losses. In what follows, we unpack both these concepts in formal notation.

6.1 Trust

Consider a collection of undesirable events, $\{E_i\}_{i=1}^n$. We can think of each event E_i as a particular "bad thing" that we would really wish not happen. We can define *trust*, T, formally as the probability that no such even occurs.

$$Tr = 1 - \Pr\{\cup_i E_i\}. \tag{6.1.1}$$

In practice, we typically talk about having trust in a particular system. A natural way to interpret that is to define such trust in terms of confidence levels that bad things don't happen. We capture this formally in the following definition.

Definition 6.1.1. We say that we have $(1-\delta)$-*trust* in a system if

$$Tr \geq 1 - \delta,$$

or, equivalently, if

$$\Pr\{\cup_i E_i\} \leq \delta.$$

It is likely to be rather difficult to reason in terms of probabilities over a union of events. To simplify this, we can apply the union bound to obtain a lower bound on trust:

$$Tr \geq 1 - \sum_i \Pr\{E_i\}.$$

Similarly, if

$$\sum_i \Pr\{E_i\} \leq \delta,$$

we have (at least) $(1-\delta)$-trust in the system. Moreover, if we ensure that $\Pr\{E_i\} \leq \delta$ for each event i, then we have $(1-n\delta)$-trust in the system, and can focus on establishing the corresponding confidence level δ for each negative event E_i in isolation.

6.2 Risk

The convenience of the definition of trust we used above is that we need not concern ourselves with determining precisely what the consequences of bad events are; instead, we can just specify the collection of things we wish to avoid, and simply focus on avoiding them. While that is at times advantageous, proper decision theoretic analysis does warrant a specification of losses, or disutilities, associated with different negative outcomes.

Let O be the outcome space, and let us partition it into two parts, O_g and O_b, corresponding to "good" and "bad" outcomes respectively. Assume that only bad outcomes correspond to actual losses, and let $C(o)$ be a loss if the outcome is $o \in O_b$. We can then define *risk* as the expected loss,

$$R = \sum_{o \in O_b} C(o)P(o),$$

where $P(o)$ is the probability that outcome is o.

6.3 Connection to Commonly Used Risk Models

There are two models for risk that are commonly used. The first defines risk as

$$R_1 = L \times C,$$

where L is likelihood of a bad event and C is consequence. The second defines risk as

$$R_2 = T \times V \times C,$$

where T is threat, V is vulnerabilities, and C is consequence.

Now, recall our expression for risk, R. Let C be the worst-case loss we can incur. Then

$$R \leq C \sum_{o \in O_b} P(o) = \Pr\{O_b\} \times C.$$

Defining $L = \Pr\{O_b\}$, we obtain R_1 as the upper bound on risk.

The second definition of risk, R_2, merely unpacks the likelihood $\Pr\{O_b\}$ as

$$\Pr\{O_b\} = \Pr\{O_b|threat\} \times \Pr\{threat\},$$

with T above being the shorthand for probability of threat (e.g., attack), $\Pr\{threat\}$, and V the shorthand for the probability of a negative outcome given that there is a threat (attack), $\Pr\{O_b|threat\}$.[1] Thus, R_2 and R_1 are equivalent expressions for the same quantity, both providing an upper bound on risk R.

6.4 Decision-Theoretic Foundations of Trust and Risk

A crucial simplification that is at the root of a decision-theoretic treatment of trust and risk is the assumption that *threats do not react to mitigation strategies*. This is the case when threats are natural disasters or unintentional human errors. However, when threats are malicious, this assumption is rather more heroic, although it can still be justified somewhat in settings where it is difficult for an attacker to observe or determine what mitigation strategies are undertaken or their consequences from attacker's perspective. Below we shall return to this issue.

In the meantime, we let M represent a decision variable that captures whatever one could do to mitigate against risk and thereby enhance trust in the system. We can expand the definition of risk to capture the result of mitigation as follows:

$$R(M) = \sum_{o \in O_b} C(o)P(o|threat, M)P(threat).$$

Thus, risk is a function of mitigation, and the mitigator aims to solve the following optimization problem:

$$\min_M R(M),$$

that is, the mitigator wishes, naturally, to minimize risk exposure.

Considering risk alone may be problematic, since mitigation strategies may well reduce functionality of the system. For example, we can considerably mitigate risk of cyber attacks by disabling internet access, but it's unlikely that such a policy is acceptable. One way to consider such costs is to explicitly invoke a cost function, $c(M)$ which captures consequences of mitigation that we also wish to minimize. Then, we would rewrite the optimization problem as

$$\min_M R(M) + c(M).$$

Arguably the most principled way to arrive at the cost function $c(M)$ is by considering positive aspects of the system, as a function of mitigation. Let $V(o)$ be the value of the system when the

[1] We thank Alison Kubota for elucidating this interpretation.

outcome is $o \in O_g$. Then the expected utility of the system, as a function of mitigation strategies, is the total gains less total losses:

$$U(M) = \sum_{o \in O_g} V(o)P(o|threat,M)P(threat) - \sum_{o \in O_b} C(o)P(o|threat,M)P(threat). \tag{6.4.1}$$

This, incidentally, captures the fact that threats may not simply cause a disaster, but may also degrade system functionality, which is a loss relative to ideally functional system, but not necessarily a net loss. Our goal is thus to maximize expected utility:

$$\max_M U(M).$$

There is an alternative decision-theoretic perspective on this which may be more natural in certain settings. Let outcome space O represent "baseline" states of the world. Assign to every outcome $o \in O$ a utility $V(o)$ if there is no successful threat and let $C(o)$ be the cost of a successful threat in state o. We can then write the expected utility as

$$U(M) = \sum_{o \in O} V(o) - C(o)P(success|threat,o,M)P(threat|o).$$

In the context of $(1-\delta)$-trust, our goal is somewhat simpler: we wish to find M such that

$$P(E_i|threat,M)P(threat) \leq \delta \quad \forall\, i = 1,\ldots,n.$$

(Recall that E_i are events we wish to avoid). From a decision theoretic perspective, we may wish to optimize expected utility subject to a constraint on trust. For example:

$$\max_M \quad U(M) \tag{6.4.2a}$$
$$\text{s.t.}:$$
$$P(E_i|threat,M)P(threat) \leq \delta \quad \forall\, i = 1,\ldots,n. \tag{6.4.2b}$$

Alternatively, we can minimize costs of mitigations, subject to a trust constraint:

$$\min_M \quad c(M) \tag{6.4.3a}$$
$$\text{s.t.}:$$
$$P(E_i|threat,M)P(threat) \leq \delta \quad \forall\, i = 1,\ldots,n. \tag{6.4.3b}$$

6.5 Game-Theoretic Foundations of Trust and Risk

Game theory takes fundamental issue with the assumption we made above that threats do not react to mitigation strategies. Realistically, there are two types of threats:

1. Unintended threats (e.g., natural disasters, accidents)

2. Malicious threats (attackers)

Game theoretic approach to trust and risk is a generalization of the decision-theoretic approach described above *which explicitly accounts for malicious threats* who in all likelihood will react to mitigation, for example, by circumventing the defenses. Fundamentally, we will explicitly distinguish the two types of threats, with T_n referring to unintended threats ("nature") and T_a referring to attackers. Let $P(T_n)$ be the probability of facing the former type of threat and $P(T_a)$ the probability of facing the latter. For simplicitly, we shall assume that at most one of these threats actually materializes. Then we can define

$$P(threat|M) = P(T_n) + P(T_a|M).$$

Note that $P(T_n)$ corresponds to $P(threat)$ that we used in the previous section and does not depend on mitigation strategies. However, $P(T_a|M)$ captures the fact that we may deter an attacker if we use mitigations properly.

This is only a small part of the story, however. The key to a game theoretic approach to trust and risk is to unpack the probabilities $P(o|T_a,M)$. In a sense this expression is completely general: the notation already captures the possibility that the attacker responds to mitigations. The problem is that it is difficult to obtain data to estimate such an attacker response function effectively. Thus, this expression is unlikely to be of practical use in capturing attackers' reasoning process. Game theory offers a principled approach to resolving this issue which posits that the attacker will respond by *selecting a strategy that is optimal for him*, given mitigations. Suppose that an attacker can choose from among a set of attacks A. Additionally, suppose that the attacker has a utility function over outcomes o, $U_a(o)$. If we fix M, the attacker will choose $a^*(M)$ such that

$$a^*(M) = \arg\max_{a \in A} U_a(o)P(o|M,a).$$

Note that every M corresponds to an optimal attacker decision $a^*(M)$ (or a set of these, in which case we can use one of a number of tie-breaking approaches, or posit, in general, a probability distribution over attacker's responses; for simplicity, we punt on these technical complications here, as we are after the conceptual points). The defender's utility function in Equation 6.4.1 from the previous section then

$$\begin{aligned} U(M,a^*(M)) = \sum_{o \in O_g} & V(o)[P(T_n)P(o|T_n,M) + P(T_a|M)P(o|T_a,M,a^*(M))] \\ & - \sum_{o \in O_b} C(o)[P(T_n)P(o|T_n,M) + P(T_a|M)P(o|T_a,M,a^*(M))]. \end{aligned}$$

We can make the same modification to the other optimization formulations in the previous section, capturing explicitly the attacker's response. As another example, consider a game theoretic generalization of the optimization problem 6.4.2:

$$\max_M \quad U(M,a^*(M)) \tag{6.5.1a}$$

s.t. :

$$P(E_i|T_n,M)P(T_n) + P(E_i|T_a,M,a^*(M))P(T_a|M) \leq \delta \quad \forall\, i = 1,\ldots,n \tag{6.5.1b}$$

$$a^*(M) = \arg\max_{a \in A} U_a(o)P(o|M,a). \tag{6.5.1c}$$

6.6 Discussion

There are a few high-level points to take away from this chapter. The first is that the way risk and trust is usually framed is extremely limited in expressiveness, as it is usually reduced to only very basic concepts, such as difficulty, likelihood, and consequence. The fact is, however, that *likelihood is endogenous*, in the sense that defense decisions (mitigations) will change it both directly (by addressing known problems) and indirectly (by deterring attacks), difficulty (to an adversary) captures too many aspects of the adversary's problem, which could include capability requirements, access requirements, resource availability, or costs, each of which can have a dramatically different impact on the attacker's decision problem and, consequently, on our exposure to risk. Finally, it is crucial to reason about the endogenous response of attackers to mitigations. Modeling an attacker explicitly allows one to appreciate the importance of understanding *attacker's motivations*. Attackers vary in motivation, depending on who the attacker is, and many threats, such as "nature", have no motivation.

Indeed, risk analysis methods do often make an explicit distinction between attacker's goals (which are a source of attack scenarios), and (defender) consequences. The distinction between goals and consequences is, however, artificial: both of these are a function of *outcomes*, i.e., what actually happens once both attacker and defender make their respective moves, with attacker's goals reflecting his own motivations (utility), while what we usually term consequences can be equivalently viewed as defender's motivations, or, perhaps, things that the defender wishes to avoid (e.g., goals with a negative value). Once this distinction is explicated in terms of *different valuations over outcomes* for the defender and attacker, and once we appreciate that the interactions between the defender and the attacker *determine the relatively likelihood of the various outcomes*, we can lay principled mathematical foundations for risk analysis and trust, using decision theory and game theory.

References

[1] M. Adler, H. Racke, N. Sivadasan, C. Sohler, and B. Vocking. Randomized pursuit-evasion in graphs. *Combinatorics, Probability and Computing*, 12:225–244, 2003.

[2] N. Agmon, N. Hazon, and G. A. Kaminka. The giving tree: Constructing trees for efficient offline and online multi-robot coverage. *Annals of Mathematics and Artificial Intelligence*, 52:143–168, 2008.

[3] N. Agmon, G. A. Kaminka, and S. Krause. Multi-robot adversarial patrolling: Facing a full-knowledge opponent. *Journal of Artificial Intelligence Research*, 42:887–916, 2012.

[4] N. Agmon, S. Kraus, and G. A. Kaminka. Multi-robot perimeter patrol in adversarial settings. In *IEEE International Conference on Robotics and Automation*, pages 2339–2345, 2008.

[5] N. Agmon, D. Urieli, and P. Stone. Multiagent patrol generalized to complex environmental conditions. In *Twenty-Fifth National Conference on Artificial Intelligence*, 2011.

[6] R. Albert, H. Jeong, and A.-L. Barabasi. Error and attack tolerance of complex networks. *Nature*, 406:378–382, 2000.

[7] A. Almeida, G. Ramalho, H. Santana, P. Tedesco, T. Menezes, V. Corruble, and Y. Chevaleyre. Recent advances in multi-agent patrolling. In *Brazilian Symposium on Artificial Intelligence*, pages 126–138, 2004.

[8] S. Alpern. Infiltration games on arbitrary graphs. *Journal of Mathematical Analysis and Applications*, 163:286–288, 1992.

[9] S. Alpern and M. Asic. The search value of a network. *Networks*, 15:229–238, 1985.

[10] S. Alpern and R. Fokkink. Accumulation games on graphs. *Networks*, 2011. to appear.

[11] S. Alpern, A. Morton, and K. Papadaki. Patrolling games. *Operations Research*, 59(5):1246–1257, 2011.

[12] B. An, J. Pita, E. Shieh, M. Tambe, C. Kiekintveld, and J. Marecki. Guards and protect: Next generation applications of security games. In *SIGECOM*, volume 10, pages 31–34, March 2011.

[13] R. J. Anderson. *Security Engineering*. Wiley, 2nd edition, 2008.

[14] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. *SIAM J. Comput.*, 38(4):1602–1623, 2008.

[15] T. August and T. I. Tunca. Who should be responsible for software security? A comparative analysis of liability policies in network environments. *Management Science*, 57(5):934–959, 2011.

[16] R. Avenhaus, B. von Stengel, and S. Zamir. Inspection games. In R. Aumann and S. Hart, editors, *Handbook of Game Theory*, pages 1947–1987. Elsevier Science Publishers, 2002.

[17] P. Bak, C. Tang, and K. Wiesenfeld. Self-organized criticality: an explanation of 1 / f noise. *Phys. Rev. Lett.*, 59(4):381–384, 1987.

[18] N. Basilico and N. Gatti. Automated abstraction for patrolling security games. In *Twenty-Fifth National Conference on Artificial Intelligence*, pages 1096–1099, 2011.

[19] N. Basilico, N. Gatti, and F. Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Eighth International Conference on Autonomous Agents and Multiagent Systems*, pages 57–64, 2009.

[20] N. Basilico, N. Gatti, and F. Amigoni. Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder. *Artificial Intelligence Journal*, 184-185:78–123, 2012.

[21] N. Basilico, N. Gatti, and F. Villa. Asynchronous multi-robot patrolling against intrusion in arbitrary topologies. In *Twenty-Forth National Conference on Artificial Intelligence*, 2011.

[22] N. Basilico, D. Rossignoli, N. Gatti, and F. Amigoni. A game-theoretic model applied to an active patrolling camera. In *International Conference on Emerging Security Technologies*, pages 130–135, 2010.

[23] B. Bosansky, V. Lisy, M. Jakov, and M. Pechoucek. Computing time-dependent policies for patrolling games with mobile targets. In *Tenth International Conference on Autonomous Agents and Multiagent Systems*, pages 989–996, 2011.

[24] G. Brown, M. Carlyle, J. Salmeron, and K. Wood. Defending critical infrastructure. *Interfaces*, 36(6):530–544, 2006.

[25] G. G. Brown, W. M. Carlyle, R. C. Harney, E. M. Skroch, and R. K. Wood. Interdicting a nuclear-weapons project. *Operations Research*, 57(4):866–877, 2009.

[26] J. M. Carlson and J. Doyle. Highly Optimized Tolerance: A mechanism for power laws in designed systems. *Phys. Rev. E*, 60(2):1412–1427, 1999.

[27] J. M. Carlson and J. Doyle. Highly Optimized Tolerance: Robustness and design in complex systems. *Phys. Rev. Lett.*, 84(11):2529–2532, 2000.

[28] J. M. Carlson and J. Doyle. Complexity and robustness. *Proc. Natl. Acad. Sci.*, 99(suppl. 1):2538–2545, 2002.

[29] H. Cavusoglo, B. Mishra, and S. Raghunathan. A model for evaluating IT security investments. *Communications of the ACM*, 47(7):87–92, 2004.

[30] H. Cavusoglu, H. Cavusoglu, and J. Zhang. Security patch management: Share the burden or share the damage. *Management Science*, 54(4):657–670, 2008.

[31] H. Cavusoglu, B. Mishra, and S. Raghunathan. The value of intrusion detection systems in information technology security architecture. *Information Systems Research*, 16(1):28–46, 2005.

[32] H. Cavusoglu and S. Raghunathan. Configuration of detection software: A comparison of decision and game theory approaches. *Decision Analysis*, 1(3):131–148, 2004.

[33] H. Cavusoglu, S. Raghunathan, and H. Cavusoglu. Configuration of and interaction between information security technologies: The case of firewalls and intrusion detection systems. *Information Systems Research*, 20(2):198–217, 2009.

[34] CERT. Frequently asked questions about the melissa virus. CERT Program at Software Engineering Institute, Carnegie Mellon University, May 1999.

[35] Y. Chevaleyre. Theoretical analysis of the multi-agent patrolling problem. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 302–308, 2004.

[36] F. Chung, J. E. Cohen, and R. Graham. Pursuit-evasion games on graphs. *Journal of Graph Theory*, 12(2):159–167, 1988.

[37] S. Clar, B. Drossel, and F. Schwabl. Forest fires and other examples of self-organized criticality. *J. Phys. Cond. Matt.*, 8(23):6803–6824, 1996.

[38] V. Conitzer and D. Korzhyk. Commitment to correlated strategies. In *Twenty-Fifth National Conference on Artificial Intelligence*, pages 632–637, 2011.

[39] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *Seventh ACM Conference on Electronic Commerce*, pages 82–90, 2006.

[40] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *Seventh ACM conference on Electronic commerce*, pages 82–90, 2006.

[41] K. J. Cormican, D. P. Morton, and R. K. Wood. Stochastic network interdiction. *Operations Research*, 46(2):184–197, 1998.

[42] M. Corporation. Common attack pattern enumeration and classification, 2012.

[43] M. Cremonini and D. Nizovtsev. Understanding and influencing attackers' decisions: Implications for security investment strategies. In *Workshop on the Economics of Information Security*, 2006.

[44] P. S. Dodds and D. J. Watts. A generalized model of social and biological contagion. *Journal of Theoretical Biology*, 232:587–604, 2005.

[45] P. Domingos. Metacost: A general method for making classifiers cost-sensitive. In *ACM International Conference on Knowledge Discovery and Data Mining*, 1999.

[46] D. P. Duggan, S. R. Thomas, C. K. K. Veitch, and L. Woodard. Categorizing threat: Building and using a generic threat matrix. Technical report, Sandia National Laboratories, 2007. SAND2007-5791.

[47] J. Edmonds. Maximum matching and a polyhedron with 0-1 vertices. *Journal of Research of the National Bureau of Standards*, 69:125–130, 1965.

[48] Y. Elmaliach, N. Agmon, and G. A. Kaminka. Multi-robot area patrol under frequency constraints. *Annals of Mathematics and Artificial Intelligence*, 57:293–320, 2009.

[49] J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer-Verlag, 1997.

[50] M. M. Flood. The hide and seek game of Von Neumann. *Management Science*, 18(5):107–109, 1972.

[51] D. Fudenberg and D. K. Levine. *The Theory of Learning in Games*. The MIT Press, 1998.

[52] S. Gal. Search games with mobile and immobile hider. *SIAM Journal of Control and Optimization*, 17(1):99–122, 1979.

[53] L. K. Gallos, F. Liljeros, P. Argyrakis, A. Bunde, and S. Havlin. Improving immunization strategies. *Physical Review E*, 75:045104, 2007.

[54] S. Ganzfried and T. Sandholm. Computing equilibria by incorporating qualitative models. In *Nineth International Conference on Autonomous Agents and Multiagent Systems*, pages 183–190, 2010.

[55] A. Glad, O. Simonin, O. Buffet, and F. Charpillet. Theoretical study of ant-based algorithms for multi-agent patrolling. In *European Conference on Artificial Intelligence*, pages 626–630, 2008.

[56] C. Gollier. *The Economics of Risk and Time*. The MIT Press, 2004.

[57] J. Grossklags, N. Christin, and J. Chuang. Secure or insure? A game-theoretic analysis of information security games. In *Seventeenth International World Wide Web Conference*, pages 209–218, 2008.

[58] E. Halvorson, V. Conitzer, and R. Parr. Multi-step multi-sensor hider-seeker games. In *Twenty-First International Joint Conference on Artificial Intelligence*, pages 159–166, 2009.

[59] C. L. Henley. Statistics of a ”self-organized” percolation model. *Phys. Rev. Lett.*, 71(17):2741–2744, 1993.

[60] V. Isler, S. Kannan, and S. Khanna. Randomized pursuit-evasion in a polygonal environment. *IEEE Transactions on Robotics*, 21(5):875–884, 2005.

[61] M. Jain, E. Kardes, C. Kiekintveld, M. Tambe, and F. Ordonez. Security games with arbitrary schedules: A branch and price approach. In *Twenty-Fourth National Conference on Artificial Intelligence*, 2010.

[62] M. Jain, D. Korzhyk, O. Vanek, V. Conitzer, M. Pechoucek, and M. Tambe. A double oracle algorithm for zero-sum security games on graphs. In *Tenth International Conference on Autonomous Agents and Multiagent Systems*, 2011.

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

[64] S. Kauffman, W. G. Macready, and E. Dickinson. Divide to coordinate: Coevolutionary problem solving. Unpublished manuscript, 1994.

[65] D. Kempe, J. M. Kleinberg, and Éva Tardos. Maximizing the spread of influence in a social network. In *Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146, 2003.

[66] C. Kiekintveld, M. Jain, J. Tsai, J. Pita, F. Ordóñez, and M. Tambe. Computing optimal randomized resource allocations for massive security games. In *Proceedings of the Eighth International Conference on Autonomous Agents and Multiagent Systems*, 2009.

[67] C. Kiekintveld, M. Jain, J. Tsai, J. Pita, F. Ordóñez, and M. Tambe. Computing optimal randomized resource allocations for massive security games. In *Seventh International Conference on Autonomous Agents and Multiagent Systems*, 2009.

[68] K. Kikuta and W. Ruckle. Continuous accumulation games on discrete locations. *Naval Research Logistics*, 49(1):60–77, 2002.

[69] D. Korzhyk, V. Conitzer, and R. Parr. Complexity of computing optimal stackelberg strategies in security resource allocation games. In *In AAAI-10*, 2010.

[70] D. Korzhyk, Z. Yin, C. Kiekintveld, V. Conitzer, and M. Tambe. Stackelberg vs. nash in security games: An extended investigation of interchangeability, equivalence, and uniqueness. *Journal of Artificial Intelligence Research*, 41:297–327, 2011.

[71] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Sixteenth Annual Conference on Theoretical Aspects of Computer Science*, pages 404–413, 1999.

[72] R. Krutz and R. D. Vines. *The CISSP Prep Guide*. Wiley Computer Publishing, 2001.

[73] H. Kunreuther and G. Heal. Interdependent security. *Journal of Risk and Uncertainty*, 26(2-3):231–249, 2003.

[74] W. Lee, M. Miller, S. Stolfo, K. Jallad, C. Park, E. Zadok, and V. Prabhakar. Toward cost-sensitive modeling for intrusion detection. *Journal of Computer Security*, 10(1/2):5–22, 2002.

[75] J. Letchford and V. Conitzer. Computing optimal strategies to commit to in extensive-form games. In *Eleventh ACM conference on Electronic commerce*, EC '10, pages 83–92, New York, NY, USA, 2010. ACM.

[76] J. Letchford and V. Conitzer. Computing optimal strategies to commit to in extensive-form games. In *Eleventh ACM Conference on Electronic Commerce*, pages 83–92, 2010.

[77] J. Letchford, L. MacDermed, V. Conitzer, R. Parr, and C. Isbell. Computing optimal strategies to commit to in stochastic games. In *Twenty-Sixth National Conference on Artificial Intelligence*, 2012.

[78] J. Letchford and Y. Vorobeychik. Computing optimal security strategies for interdependent assets. In *Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 2012.

[79] G. McCormick. Computability of global solutions to factorable nonconvex programs: Part I - convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[80] J. C. Miller and J. M. Hyman. Effective vaccination strategies for realistic social networks. *Physica A*, 386:780–785, 2007.

[81] J. Mounzer, T. Alpcan, and N. Bambos. Integrated security risk management for IT-intensive organizations. In *Sixth International Conference on Information Assurance and Security*, pages 329–334, 2010.

[82] M. V. Nehme. *Two-Person Games for Stochastic Network Interdiction: Models, Methods, and Complexities*. PhD thesis, The Unversity of Texas at Austin, 2009.

[83] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[84] M. Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[85] M. E. J. Newman, M. Girvan, and J. D. Farmer. Optimal design, robustness, and risk aversion. *Phys. Rev. Lett.*, 89(2):028301, 2002.

[86] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, editors. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[87] U. of Oregon Route Views Project. Online data and reports. http://www.routeviews.org.

[88] H. Ogut, H. Cavusoglu, and S. Raghunathan. Intrusion-detection policies for IT security breaches. *INFORMS Journal on Computing*, 20(1):112–123, 2008.

[89] H. Ogut, N. Menon, and S. Raghunathan. Cyber insurance and IT security investments: Impact of interdependent risk. In *Workshop on the Economics of Information Security*, 2005.

[90] T. Parsons. Pursuit-evasion in a graph. *Lecture Notes in Mathematics: Theory and Applications of Graphs*, 642:426–441, 1976.

[91] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordóñez, and S. Kraus. Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games. In *Proceedings of the Seventh International Conference on Autonomous Agents and Multiagent Systems*, pages 895–902, 2008.

[92] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordonez, and S. Kraus. Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games. In *Proc. of The 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 895–902, 2008.

[93] R. Pastor-Satorras and A. Vespignani. Immunization of complex networks. *Physical Review E*, 65:036104, 2002.

[94] J. Pita, M. Jain, F. Ordóñez, C. Portway, M. Tambe, C. Western, P. Paruchuri, and S. Kraus. Using game theory for los angeles airport security. *AI Magazine*, 30(1):43–57, 2009.

[95] F. Provost and T. Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *KDD*, pages 43–48, 1997.

[96] ROADMAP. Roadmap to achieve energy delivery systems cybersecurity. Energetics, Inc, September 2011.

[97] B. Roberson. The colonel Blotto game. *Economic Theory*, 29:1–24, 2006.

[98] L. Rosencrance. Melissa virus author sentenced. PC World, May 2002.

[99] T. Roughgarden. *Selfish Routing and the Price of Anarchy*. The MIT Press, 2005.

[100] T. C. Shelling. *The Strategy of Conflict*. Harvard University Press, 1981.

[101] E. Shieh, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, and G. Meyer. PROTECT: A deployed game theoretic system to protect the ports of the United States. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems*, pages 13–20, 2012.

[102] K. Soramaki, M. L. Bech, J. Arnold, R. J. Glass, and W. Beyeler. The topology of interbank payment flows. *Physica A*, 379:317–333, 2007.

[103] J. E. Stamp, R. A. Laviolette, L. R. Phillips, and B. T. Richardson. Final report: Impacts analysis for cyber attack on electric power systems. Sandia National Laboratories Technical Report, SAND2009-1673, February 2009.

[104] T.J. Lambert III, M. Epelman, and R. L. Smith. A fictitious play approach to large-scale optimization. *Oper. Res.*, 53(3):477–489, 2005.

[105] J. Tsai, T. H. Nguyen, and M. Tambe. Security games for controlling contagion. In *Twenty-Sixth National Conference in Artificial Intelligence*, 2012. to appear.

[106] J. Tsai, Z. Yin, J. young Kwak, D. Kempe, C. Kiekintveld, and M. Tambe. Urban security: Game-theoretic resource allocation in networked physical domains. In *Twenty-Fourth National Conference on Artificial Intelligence*, 2010.

[107] J. W. Ulvila and J. E. Gaffney. A decision analysis method for evaluating computer intrusion detection systems. *Decision Analysis*, 1(1):35–50, 2004.

[108] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 60 anv edition, 2007.

[109] B. von Stengel and S. Zamir. Leadership games with convex strategy sets. *Games and Economic Behavior*, 69(2):446–457, 2010.

[110] Y. Vorobeychik and M. P. Wellman. Stochastic search methods for Nash equilibrium approximation in simulation-based games. In *Seventh International Conference on Autonomous Agents and Multiagent Systems*, pages 1055–1062, 2008.

[111] R. K. Wood. Deterministic network interdiction. *Mathematical Computer Modelling*, 17(2):1–18, 1993.

[112] D. L. Woodruff, editor. *Network Interdiction and Stochastic Integer Programming*. Kluwer Academic Publishers, 2003.

[113] W. T. Yue and A. Bagchi. Tuning the quality parameters of a firewall to maximize net benefit. In *International Workshop on Distributed Computing*, pages 321–329, 2003.

[114] T. Zhou, J. M. Carlson, and J. Doyle. Evolutionary dynamics and highly optimized tolerance. *J. Theor. Biol.*, 236:438–447, 2005.

[115] J. Zhuang and V. Bier. Balancing terrorism and natural disasters—Defensive strategy with endogenous attacker effort. *Operations Research*, 55(5):976–991, 2007.

DISTRIBUTION:

1	MS 0899	Technical Library, 9536 (electronic copy)
1	MS 0123	D. Chavez, LDRD Office, 1011

Sandia National Laboratories