Human Adversaries in Security Games: Integrating Models of Bounded Rationality and Fast Algorithms

by

Rong Yang

A Dissertation Presented to the
FACULTY OF THE USC GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(Computer Science)

April 2014

Acknowledgments

Five years ago, I decided to join USC pursing a PhD degree. This was one of the most important decisions I have made in my life. Finishing the PhD wasn't easy, but I have really enjoyed the past five years with the privilege to have worked with a number of extraordinary people, who have given me invaluable advices on both research and life.

First of all, I would like to give my special thanks to my PhD advisor Milind Tambe, without whom I would not been anywhere near where I am now. When I first joined the TEAMCORE research group, little did I know about what it means to do research. Milind, with his endless patience, guided me through each step to do meaning research. He taught me the importance of conducting research that has real-world impact. His dedication for his students has encouraged us to not only do better research but also be a better person. His unbounded passion for research and unbeatable working attitude has been and will always be the inspiration to me. I will always remember the days when he worked with me until the last minute on the paper deadlines! I am also grateful for his support as I started my family during the course of pursing my PhD. His understanding and consideration has made the toughest time of my life so much smoother. Milind, thanks for always being there when I needed a discussion regarding research, a break to take care of my baby, and a heartfelt advice for future career path.

Next, I would like to thank other members of my thesis committee: Fernando Ordóñez, Jonathan Gratch, Rajiv Maheswaran, Richard John, and Vincent Conitzer, for providing valuable feedback to my research and pushing me to think deeper. As my research being at the intersection of many disciplines, I would never be able to push my research to the height I have achieved without having an interdisciplinary committee to help shaping my idea. A special thanks to Fernando, for his tremendous guidance to me and ceaseless confidence in me, without what my research would have suffered greatly. You introduced me to the world of large-scale optimization techniques. I will always remember the days when I could just walk into your office to ask questions whenever I encountered problems in my research. Thanks for flying over all the way from Chili to support my research. Richard, thanks for opening up the door to human subject research for me. Without your guidance, I would never be able to reach the achievement in my research. I really appreciate your always bringing a different perspective to my research. As a computer scientist, I enjoy interacting with human subjects in my experiment as much as I do with my codes, thanks to your help.

I would also like to thank the many excellent researchers that I have had the privilege to work with over the years. This list includes Christopher Kiekintveld, Matthew Taylor, Bo An, Albert Xin Jiang. James Pita, Jun-young Kwak, Sarit Kraus, Thanh Nguyen, Fei Fang, Benjamin Ford and Debrun Kar. I thank all the students who helped develop the games for my experiment: Mayuresh Janorkar, Mohit Goenka, Karthik Rajagopal and Noah Olsman. I am also grateful for the support from the Army Research Office, the US Coast Guard and IBM on my research over the years. They have provided me not only the opportunity to work on real-world problems of research interest, but also the possibility to develop my own research interest. I would like to

particularly thank Janusz Marecki for his help in applying for the IBM fellowship and for being the best academic brother I could ever ask for.

During my time at USC, I enjoyed myself very much as a member of the large TEAMCORE family. I appreciate the time shared with those during my PhD career: Matthew Brown, Leandro Marcolino, Chao Zhang, Yundi Qian, Kumar Amulya Yadav, Haifeng Xu, Francesco Delle Fave and William Haskell. Special thanks to Manish Jain for all the advices you have given me over the years; Jason Tsai for being a great officemate with the patience to listen to my stories from shopping for shoes to taking care of my baby boy; Eric Shieh for the delicious food you have brought to the office; Zhengyu Yin for the many afternoons that you have spend discussing with me about research; Jun-young Kwak for always having the good recommendations for Korean restaurants.

Finally, I would like to thank my family for their support over the years. In particular, thanks to my parents for always believing in me, for supporting my decision of flying aboard to pursue my career, and for coming over to the US to help take care of Dylan. I would also like to thank my in-laws for the support during the toughest time of my PhD career. Lastly, I would like to thank my dear husband Qing for always being there for me through happiness and toughness, for being the best friend of my life, for supporting me to pursue my own career and for bringing Dylan, the most precious gift ever, to my life, .

Table of Contents

Acknowledgments **ii**

List of Figures **ix**

Abstract **xi**

Chapter 1: Introduction **1**
1.1 Problem Addressed . . . 2
1.2 Contributions . . . 4
1.2.1 Stochastic Models of Adversary Decision Making . . . 4
1.2.2 Algorithms for Optimizing Defender Strategy . . . 5
1.2.3 Adaptive Resource Allocation and Application for Protecting Wildlife . . . 7
1.3 Overview of Thesis . . . 8

Chapter 2: Background **10**
2.1 Stackelberg Games . . . 10
2.1.1 Bayeisan Stackelberg Games . . . 12
2.1.2 Strong Stackelberg Equilibrium . . . 13
2.1.3 Stackelberg Security Games . . . 14
2.2 Los Angeles International Airport . . . 16
2.3 Baseline Solvers . . . 18
2.3.1 Defender Optimal Strategy against a perfectly rational adversary . . . 19
2.3.2 Defender Optimal Strategy against the $\epsilon-$optimal adversary response . . . 20
2.4 Human Subject Experiments . . . 21

Chapter 3: Related Work **24**
3.1 Behavioral Game Theory . . . 24
3.2 Efficient Computation of Defender Optimal Strategy . . . 26
3.3 Robust Defender Strategies . . . 28
3.4 Learning Adversary Behavioral in Repeated Games . . . 29

Chapter 4: Modeling Adversary Decision Making **30**
4.1 Models for Predicting Attacker Behaviors . . . 31
4.1.1 Prospect Theory . . . 32
4.1.2 Quantal Response . . . 33

4.1.3 Quantal Response with Rank-related Expected Utility . . . 35
4.2 Computing Optimal Defender Strategy . . . 36
4.2.1 Computing against a PT-adversary . . . 36
4.2.1.1 BRPT . . . 36
4.2.1.2 RPT . . . 41
4.2.2 Computing against a QR-adversary . . . 42
4.2.3 Computing against a QRRU-adversary . . . 43
4.3 Parameter Estimation . . . 45
4.3.1 Selecting Payoff Structures . . . 46
4.3.2 Parameter Estimation for Prospect Theory . . . 49
4.3.3 Parameter Estimation for the QR Model . . . 51
4.3.4 Parameter Estimation for the QRRU Model . . . 52
4.4 Experimental Results and Discussion . . . 54
4.4.1 A Simulated Online SSG . . . 54
4.4.2 Experimental Settings . . . 55
4.4.3 Algorithm Parameters . . . 57
4.4.4 Quality Comparison . . . 59
4.4.4.1 Average Performance . . . 60
4.4.4.2 Performance Distribution . . . 66
4.4.5 Model Prediction Accuracy . . . 69

Chapter 5: Quantal Response Model with Subjective Utility **74**
5.1 The SUQR Model . . . 75
5.1.1 Learning SUQR Parameters . . . 76
5.1.2 Prediction Accuracy of SUQR model . . . 77
5.2 Improving MATCH . . . 78
5.2.1 Selecting β for MATCH: . . . 80
5.3 Experimental Results . . . 80
5.3.1 Results with AMT Workers, 8-target Games . . . 81
5.3.2 SU-BRQR vs MATCH . . . 82
5.3.3 SU-BRQR vs Improved MATCH . . . 83
5.3.4 Results with New Experimental Scenarios . . . 84
5.3.4.1 Security Intelligence Experts, 8-target games . . . 85
5.3.4.2 SU-BRQR vs DOBSS . . . 85
5.3.4.3 SU-BRQR vs MATCH . . . 85
5.3.5 Bounded Rationality of Human Adversaries . . . 86
5.3.6 AMT Workers, 24-target Games . . . 87
5.3.6.1 SU-BRQR vs MATCH with Parameters Learned from the 8-target Games . . . 87
5.3.6.2 SU-BRQR vs DOBSS with Re-estimated Parameters . . . 88
5.3.6.3 SU-BRQR vs MATCH with Re-estimated Parameters . . . 88

Chapter 6: Modeling Human Adversaries in Network Security Games **90**
6.1 Problem Definition . . . 91
6.2 Adversary Model . . . 94
6.2.1 Basic Quantal Response Model . . . 95
6.2.2 Quantal Response with Heuristics . . . 95
6.3 Model Parameter Estimation . . . 96
6.3.1 Data Collection . . . 96
6.3.2 Training the QR Model . . . 99
6.3.3 Training the QRH Model . . . 100
6.4 Computing Defender Resource Allocation Strategy . . . 102
6.4.1 Best Response to QR model . . . 103
6.4.2 Best Response to QRH model . . . 104
6.5 Experiment Results . . . 105
6.5.1 Experiment Settings . . . 105
6.5.2 Experiment Results . . . 107

Chapter 7: Computing Defender Optimal Strategy **111**
7.1 Problem Definition . . . 112
7.1.1 Resource Assignment Constraint . . . 113
7.2 Binary Search Method . . . 113
7.3 GOSAQ . . . 116
7.3.1 GOSAQ with No Assignment Constraint . . . 117
7.3.2 GOSAQ with Assignment Constraints . . . 119
7.4 PASAQ . . . 120
7.4.1 PASAQ with No Assignment Constraint . . . 122
7.4.2 PASAQ With Assignment Constraints . . . 125
7.5 Experiments . . . 126
7.5.1 No Assignment Constraints . . . 127
7.5.2 With Assignment Constraints . . . 129

Chapter 8: Scaling-up **133**
8.1 Generalized PASAQ . . . 134
8.2 COCOMO– A Branch-and-Price Algorithm . . . 137
8.3 BLADE– A Cutting-Plane Algorithm . . . 140
8.3.1 Master . . . 141
8.3.2 Separation Oracle . . . 142
8.3.3 wBLADE . . . 146
8.3.4 Quality and Runtime Trade-off . . . 148
8.4 Experimental results . . . 149

Chapter 9: Adaptive Resource Allocation and its Application to Wildlife Protection **153**
9.1 Domain . . . 153
9.2 Model in PAWS . . . 156
9.2.1 Stackelberg Game Formulation . . . 156
9.2.2 Behavioral Heterogeneity . . . 159
9.2.3 Adapting Patrolling Strategy using Historical Crime Data . . . 161

9.3 Research Advances in PAWS 162
9.3.1 Learn the Behavioral Model 162
9.3.1.1 Learning with the Identified Data 163
9.3.1.2 Learning with the Anonymous Data 164
9.3.1.3 Combining the Two Kinds of Data 165
9.3.2 Adapting Patrolling Strategy 167
9.4 Evaluation 168
9.4.1 General Game Settings 168
9.4.2 Results for the Deployment Area 170

Chapter 10: Conclusion **174**
10.1 Contributions 175
10.2 Future Work 178

Bibliography **181**

Appendix A: Error Bound of PASAQ **188**

List of Figures

1.1 US Coast Guard at the port of Boston 6

1.2 QENP: The intended site of deployment. Ranger photo taken by Andrew Lemieux. 8

2.1 LAX Security 17

4.1 Prospect Theory empirical function forms 32

4.2 Piecewise approximation of the weighting function 40

4.3 Payoff Structure Clusters (color) 47

4.4 Game interface for our simulated online SSG 54

4.5 Defender average expected utility achieved by different strategies 59

4.6 Defender average expected utility (normalized between 0 and 1) achieved by different strategies 61

4.7 Defender average expected utility achieved by QR model based strategies 64

4.8 Distribution of defender's expected utility against each individual subject 67

4.9 Distribution of defender's expected utility against each individual subject 68

6.1 Game Interface (colored) 97

6.2 Graphs Tested in Data Collection 98

6.3 Graphs Tested in Evaluation Experiments 106

6.4 Average Defender Expected Utility 109

6.5 Average Defender Expected Utility 110

7.1 Piecewise Linear Approximation . . . 120

7.2 Solution Quality and Runtime Comparison, without assignment constraints (better in color) . . . 131

7.3 Solution Quality and Runtime Comparison, with assignment constraint (better in color) . . . 132

8.1 Branching Tree . . . 137

8.2 Minimizing weighted 1-norm distance . . . 147

8.3 Runtime Comparison of the BLADE family . . . 150

8.4 Comparing COCOMO and BLADE, QR Model . . . 151

8.5 Runtime Comparison, QR-Sigmoid model . . . 152

9.1 Lioness photo courtesy of John Coppinger, Remote Africa Safaris Ltd. Poacher snare photo taken by Andrew Lemieux. . . . 155

9.2 Empirical Marginal PDF of the SUQR parameter among all the 760 subjects . . . 160

9.3 Simulation results over round . . . 168

9.4 Slow Capture v.s. Fast Capture . . . 169

9.5 Comparing cumulative EU at round 20 . . . 170

9.6 The QENP area of interest for our simulation . . . 172

9.7 Simulation results over round for the 64 sq. km grid area in QENP . . . 173

9.8 Patrolling coverage density in the park . . . 173

Abstract

Security is a world-wide concern in a diverse set of settings, such as protecting ports, airport and other critical infrastructures, interdicting the illegal flow of drugs, weapons and money, preventing illegal poaching/hunting of endangered species and fish, suppressing crime in urban areas and securing cyberspace. Unfortunately, with limited security resources, not all the potential targets can be protected at all times. Game-theoretic approaches — in the form of "security games" — have recently gained significant interest from researchers as a tool for analyzing real-world security resource allocation problems leading to multiple deployed systems in day-to-day use to enhance security of US ports, airports and transportation infrastructure. One of the key challenges that remains open in enhancing current security game applications and enabling new ones originates from the perfect rationality assumption of the adversaries — an assumption may not hold in the real world due to the bounded rationality of human adversaries and hence could potentially reduce the effectiveness of solutions offered.

My thesis focuses on addressing the human decision-making in security games. It seeks to bridge the gap between two important subfields in game theory: algorithmic game theory and behavioral game theory. The former focuses on efficient computation of equilibrium solution concepts, and the latter develops models to predict the behaviors of human players in various

game settings. More specifically, I provide: (i) the answer to the question of which of the existing models best represents the salient features of the security problems, by empirically exploring different human behavioral models from the literature; (ii) algorithms to efficiently compute the resource allocation strategies for the security agencies considering these new models of the adversaries; (iii) real-world deployed systems that range from security of ports to wildlife security.

Chapter 1: Introduction

Security is a world-wide concern in a variety of different settings, including protecting critical infrastructures such as port, airports and flights, interdicting illegal flow of drugs, weapons and money, preventing illegal poaching/hunting of endangered species and fish, suppressing crime in urban areas and securing cyberspace. The key challenge in these various security settings is that there are only limited amount of resources, therefore not all the potential targets can be protected at any time. At the same time, the adversaries are conducting surveillance, hence any deterministic allocation of the resource may be exploited by these intelligent adversaries. The security agencies often prefer to allocate their resources in a randomized fashion.

Game-theoretic approaches have recently gained significant interest from researchers as a tool for analyzing real-world security resource allocation problems [Gatti, 2008a; Agmon et al., 2008; Basiloco et al., 2009]. These models provide a sophisticated approach for generating unpredictable, randomized strategies that mitigate the ability of attackers to find weaknesses using surveillance. The ARMOR [Pita et al., 2008], IRIS [Tsai et al., 2009] and GUARDS [Pita et al., 2011] are notable examples of real-world applications. At the heart of these applications is the

leader-follower Stackelberg game model, where the leader (security forces) acts first by committing to a mixed-strategy; the follower (attacker/adversary) observes the leader's strategy and responds to it.

1.1 Problem Addressed

One of the key assumptions in the existing real-world security systems is about how attackers choose strategies based on their knowledge of the security strategy. Typically, such systems apply the standard game-theoretic assumption that attackers are perfectly rational and strictly maximize their expected utilities. This is a reasonable starting point for the first generation of deployed systems. However, in real-world security problems, the security forces are facing human adversaries whose decisions may be governed by their bounded rationality [Simon, 1956], which may lead them to deviate from the optimal choice. Hence, defense strategies based on the perfect rationality assumption may not be robust against attackers using different decision procedures. Such assumptions also fail to exploit known weaknesses in the decision-making of human attackers. Indeed, it is widely accepted that standard game-theoretic assumptions of perfect rationality are not ideal for predicting the behavior of humans in multi-agent decision problems [Camerer et al., 2004; Wright and Leyton-Brown, 2010]. Thus, it is critical to integrate more realistic models of human decision-making for solving real-world security problems.

There are several open questions we need to address in moving beyond perfect rationality assumptions. First, a large variety of alternative models have been studied in behavioral game theory and cognitive psychology [Camerer et al., 2004; Costa-Gomes et al., 2001] that capture

some of the deviations of human decisions from perfect rationality. However, there is an important empirical question of which model best represents the salient features of human behavior in applied security contexts. Given that many of these models are descriptive, integrating any of the proposed models into a decision-support system (even for the purpose of empirically evaluation) requires developing computational efficient representation of them.

Furthermore, many of these models imply mathematically complex representations of the adversary's decision-making procedure (e.g., nonlinear and non-convex function forms), which in general leads to an NP-hard problem of computing the defender's optimal strategy. Therefore, developing efficient algorithms to solve such a computationally complex problem is critical for real-world security problems due to their massiveness.

The third open question originated from domains where actual adversary events occur often and generate significant amounts of collectible crime event data, such as preventing illegal poaching of wildlife. As a result, learning behavioral models from collected adversary data and addressing the heterogeneity among large populations of adversaries becomes the new challenges in these domains. In particular, crime data can be anonymous, or it can be linked to confessed adversaries (i.e., identified). While the latter type of data provides rich information about individual adversaries, that type of data is sparse and hard to collect. The majority of collected data is evidence on crimes committed by anonymous adversaries. Compared to identified data, anonymous data provides no information about the identity of the adversary that committed the crime and therefore cannot be used to build accurate behavioral models on the individual level. The open question here is how to utilize both types of data to build and learn a better model of the large population of criminals. Moreover, how does the learned model help better predict future crime events and thus help law enforcement officials to improve their resource allocation strategies?

1.2 Contributions

My thesis will address these open questions to improve the security resource allocation strategies against human adversaries in real-world security problems.

1.2.1 Stochastic Models of Adversary Decision Making

I first investigate different theories in the behavioral literature to develop models of human decision-making for predicting adversary behavior. More specifically, I have explored two fundamental theories, i.e., Prospect Theory [Kahneman and Tvesky, 1979] and Quantal Response (QR) Model [McKelvey and Palfrey, 1995], to model the decision-making process of human adversaries [Yang et al., 2011, 2013b] through experiment with human subjects using a simulated security game that I developed. Prospect Theory is an important theory in the literature which has led Kahneman win the Nobel Prize in Economic Sciences. It provides a descriptive model of human decision making. Quantal Response Model originates from the literature of discrete choice models [Train, 2003; McFadden, 1984], which models the player's behavior as a stochastic choice making. In experiments with human subjects, the defender strategy computed using quantal response model to predict the human adversary significantly outperformed its competitors, including the previous leading contender COBRA [Pita et al., 2010].

I then further extend the QR model from three different perspectives. I first modified the QR model by replacing the expected utility with a more generalized utility function – rank-dependent utility [Yang et al., 2013b]. The rank-dependent utility function incorporates the fact that individuals overweight the low-probability event into the model. It improves the performance of the

original quantal response model in cases where the defender has potential large damage on targets covered with very few resources. I also apply the QR model to the network security games. In a network security games, the computation of actual expected utility of each action becomes very complicated for the adversary. I have discovered that extending the expected utility function with a set of easy-to-compute features would improve the performance of the model significantly [Yang et al., 2012a]. Finally, I integrate the QR model with a novel subjective utility function, which is learned from the data collected from experiments with human subjects [Nguyen et al., 2013][1]. The subjective utility function captures the fact that humans put more weight on the probability of a success attack in their decision making process. Compared with the classic Quantal Response model, the new model is shown to provide better predictions on the behavior of human adversaries. Through extensive experiments with 547 human subjects playing 11102 games in total, I emphatically answer the question of "Is there then any value in using data-driven method to model human behavior in solving SSGs?" in the affirmative.

1.2.2 Algorithms for Optimizing Defender Strategy

Given the non-convexity of mathematical model for predicting adversary behavioral, the problem for computing defender optimal strategies is also non-convex which is in general an NP-hard problem [Vavasis, 1995]. To that end, I have provided two novel algorithms (GOSAQ and PASAQ) to solve the problem [Yang et al., 2012b]. These two novel algorithms are based on three key ideas: (i) use of a binary search method to solve the fractional optimization problem efficiently, (ii) construction of a convex optimization problem through a non-linear transformation,

[1] This is a work that I co-authored with Thanh Nguyen, who is the first author.

Figure 1.1: US Coast Guard at the port of Boston

(iii) building a piecewise linear approximation of the non-linear terms in the problem. I also provided proofs of approximation bounds, detailed experimental results showing the advantages of GOSAQ and PASAQ in solution quality over the benchmark algorithm (BRQR) and the efficiency of PASAQ. Given these results, PASAQ is at the heart of the first version PROTECT [Shieh et al., 2012] system used by the US Coast Guard in the port of Boston for generating optimal patrolling strategies[2].

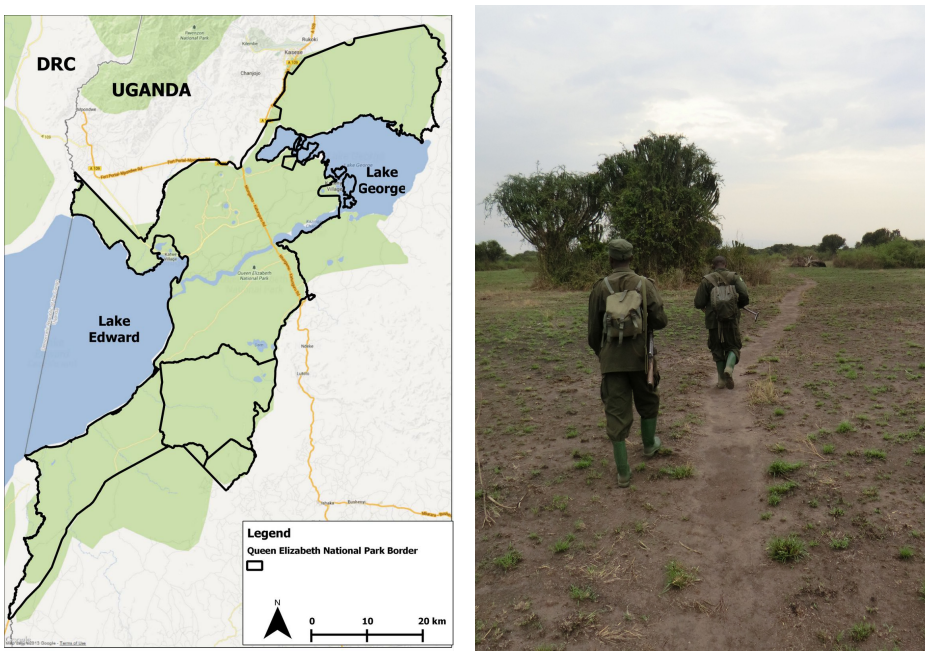
Given that many real-world security problems are massive, such as for Federal Air Marshals [Kiekintveld et al., 2009], further scaling-up for computing defender strategy incorporating models of adversary bounded rationality is needed. Unfortunately, previously proposed branch-and-price approaches fail to scale-up given the non-convexity of such models, as we show with a realization called COCOMO. Therefore, I present a novel cutting-plane algorithm called BLADE [Yang et al., 2013a] to scale-up SSGs with complex adversary models, with three novelties: (i)

[2]Since then, newer algorithms have been developed as will be discussed below

an efficient scalable separation oracle to generate deep cuts; (ii) a heuristic that uses gradient to further improve the cuts; (iii) techniques for quality-efficiency tradeoff

1.2.3 Adaptive Resource Allocation and Application for Protecting Wildlife

To address the challenges of learning adversary behavioral models from history crime data, I present the Protection Assistant for Wildlife Security (PAWS) application - a joint deployment effort done with researchers at Ugandas Queen Elizabeth National Park (QENP) with the goal of improving wildlife ranger patrols [Yang et al., 2014]. First, we propose a stochastic behavioral model which extends the current state-of-the-art to capture the heterogeneity in the decision making process of a population of poachers. Second, we demonstrate how to learn the behavioral pattern of the poacher population from both the identified data and the anonymous data. Third, in order to overcome the sparseness of the identified data, we provide a novel algorithm, PAWS-Learn, to improve the predicating accuracy of the estimated behavioral model by combining the two types of data. Fourth, we develop a new algorithm, PAWS-Adapt, which adapts the rangers' patrolling strategy against the poacher population's behavioral model. Fifth, we show the effectiveness of PAWS in a general setting, but our main drive is to deploy PAWS in QENP; we also demonstrate PAW's effectiveness when applied to an area of QENP. Our experiment results and corresponding discussion illustrate the capabilities of PAWS and its potential to improve the efforts of wildlife law enforcement officials in managing and executing their anti-poaching patrols.



(a) Outline of QENP (b) QENP rangers on patrol.

Figure 1.2: QENP: The intended site of deployment. Ranger photo taken by Andrew Lemieux.

1.3 Overview of Thesis

This thesis is organized as follows. Chapter 2 introduces the necessary background materials for the research presented in this thesis. Chapter 3 provides an overview of the related work. Chapter 4 discusses how the models for predicting the adversary decision making are developed from applying existing literature on general human behavior to security games. Chapter 5 presents an extension of the existing quantal response model to further improve its performance in predicting adversary behavior in security games. Chapter 6 investigates the performance quantal response model in the network security games. Chapter 7 explains the algorithms for efficiently computing the optimal defender strategy incorporating the behavioral model of the adversary. Chapter 8 provides further scaling-up of the computation of defender strategy for massive real-world security

problems. Chapter 9 describes the real-world application for preventing wildlife crimes. Finally, Chapter 10 concludes the thesis and presents possible future directions.

Chapter 2: Background

The work in this thesis is based on using Stackelberg game to model the security scenarios. As such, I will first outline the relevant background in Section 2.1 by introducing the general Stackelberg game model (Section 2.1.1), the Bayesian extension (Section 2.1.2), the standard solution concept knows as Strong Stackelberg Equilibrium (Section 2.1.3), and a restricted class of Stackelberg game referred to as security games (Section 2.1.4). In Section 2.2, I will describe a real-world security problem at the Los Angeles International Airport (LAX) which motives the experiment setup in this thesis. Section 2.3 overviews previous algorithms of relevance to this thesis. Finally, Section 2.4 provides the justification of conducting experiment with human subjects as an approach for evaluating the models and algorithms for this thesis.

2.1 Stackelberg Games

There are two types of players in a Stackelberg game, the leader and the follower. The leader commits to a strategy first; and then the follower responds after observing the leader's action by maximizing his utility [von Stackelberg, 2011]. For the reminder of this thesis, I will refer to the leader as 'her' and the follower as 'he' for explanatory purpose. In order to show the advantage of being the leader in a Stakelberg game, let's look at an example which was first presented by

[Conitzer and Sandholm, 2006]. Table 2.1 shows the payoff matrix of the game, where the leader is the row player and the follower is the column player. If the players move simultaneously, the only pure-strategy Nash Equilibrium for this game is when the leader plays l_a and the follower plays f_a, which gives the leader a payoff of 2. In fact, playing l_b is strictly dominated by playing l_a for the leader. However, if the row player moves first, she can commit to playing l_b, which will give her a payoff of 3 since the column player will play f_b to ensure a higher payoff. Furthermore, if the leader commits to a mixed strategy of playing l_a and l_b with equal probability (0.5), then the follower will play f_b, leading to a payoff of 3.5 for the leader.

	f_a	f_b
l_a	2,1	4,0
l_b	1,0	3,1

Table 2.1: Example of a Stackelberg game

Let Θ denote the leader and Ψ denote the follower in a Stackelberg game. Each player has a set of *pure strategies* they can play, denoted as $\sigma_\Theta \in \Sigma_\Theta$ and $\sigma_\Psi \in \Sigma_\Psi$. A *mixed strategy* allows the player to play probabilistically over a subset of the *pure strategies*. We denote the mixed strategy of the leader by $\boldsymbol{x}$. In a general Stackelberg game, $\boldsymbol{x}$ is a N-dimensional vector, where N is the number of pure strategies for the leader. The i^{th} element of $\boldsymbol{x}$, x_i, represents the probability that the leader will play pure strategy i. For the purpose of computing the equilibria, it is sufficient to consider only pure strategy response of the follower takes a pure strategy, as show in [Conitzer and Sandholm, 2006]

The payoffs for each player are defined over all possible joint pure-strategy outcomes. More formally, we define the payoff matrix for both players:

$$U = (\boldsymbol{\mu}_1 \dots \boldsymbol{\mu}_J), V = (\boldsymbol{\nu}_1 \dots \boldsymbol{\nu}_J).$$

The vector $\boldsymbol{\mu}_j$ presents the payoffs for the defender when the follower plays pure strategy j. Similarly, the vector $\boldsymbol{\nu}_j$ represents the follower's payoffs by playing pure strategy j. Given a leader's mixed strategy $\boldsymbol{x}$, the follower maximizes his expected utility by choosing one of his pure strategies. For each pure strategy j chosen by the follower, the expected utility for the leader by taking the mixed strategy $\boldsymbol{x}$ is a linear function of $\boldsymbol{x}$: $\boldsymbol{\mu}_j^T \boldsymbol{x}$. At the same time, the follower gets an expected utility of $\boldsymbol{\nu}_j^T \boldsymbol{x}$.

2.1.1 Bayeisan Stackelberg Games

In a Bayesian Stackelberg game, both players are extended to have multiple types. In this thesis, we consider only one type of the defender (leader) who is trying to allocating her resources. However, there can be multiple types of attackers (followers). For example, a security force may be interested in both protecting against potential terrorist attacks and catching drug smugglers. Each type of the follower has his own payoff matrix as well as the payoff matrix for the leader. Formerly, let $1 \leq \lambda \leq \Lambda$ denote the types of the followers. The defender faces attacker type λ with a priori probability of p^λ. The associated payoff matrix for the leader and type λ attacker is represented by (U^λ, V^λ) respectively.

Given the payoff matrix for each type, the leader commits to a mixed strategy $\boldsymbol{x}$ knowing the priori probability distribution of all different follower types but not the exact type of the follower she faces. The follower, on the other hand, knows his own type λ and plays his best response by maximizing his expected utility according to his payoff matrix V^λ, after observing the leader's mixed strategy.

The leader's goal is to maximizing her expected utility, given the priori probability distribution of all follower types and the payoff matrix. Let vector $\boldsymbol{j} = \langle j^1, ..., j^\Lambda \rangle$ denote the pure

strategies played by all types of the followers, with j^λ representing the pure strategy played by follower type λ. The leader's expected utility by playing mixed-strategy $\boldsymbol{x}$ is then defined as $\mu(\boldsymbol{x}, \boldsymbol{j}) = \sum_{\lambda=1}^{\Lambda} p^\lambda \boldsymbol{\mu}^\lambda \mu(\boldsymbol{x}, j^\lambda)$, where $\mu(\boldsymbol{x}, j^\lambda) = \mu_{j^\lambda}^T \boldsymbol{x}$ is the leader's expected utility when facing the follower of type λ.

2.1.2 Strong Stackelberg Equilibrium

The most common solution concept in game theory is a Nash equilibrium, which is a profile of strategies for each player in which no player can gain by changing only their only strategy. In other words, each player plays his/her best-response assuming that the other players also best respond by maximizing his/her expected utility. In a Stackelberg game, the solution concept that is mostly adopted is the Strong Stackelberg Equilibrium (SSE). Besides the mutual best-response feature that is entailed by Nash equilibrium, SSE also assumes that the adversary will break ties in favor of the defender. Most of the existing algorithms for solving Stackelberg security games adopt the concept of SSE [Paruchuri et al., 2008; Kiekintveld et al., 2009]. This is because that a SSE always exists in all Stackelberg games [Breton et al., 1988]. Furthermore, when ties exist, the leader can always induce the desired outcome by selecting a strategy arbitrarily close to the SSE strategy, against which the follower strictly prefers the desired strategy [von Stengel and Zamir, 2004]. Formally a Strong Stackelberg Equilibrium is defined below:

Definition 1. *For a given Bayesian Stackelberg game with utility matrices* $(U^1, V^1), \ldots, (U^\Lambda, V^\Lambda)$ *and type distribution* $\mathbf{p}$*, a pair of strategies* $(\boldsymbol{x}, \boldsymbol{j})$ *forms a Strong Stackelberg Equilibrium if and only if:*

1. *The leader plays a best response:*

   $u(\boldsymbol{x}, \boldsymbol{j}(\boldsymbol{x})) \geq u(\boldsymbol{x}', \boldsymbol{j}(\boldsymbol{x}')), \forall \boldsymbol{x}'$.

2. *The follower plays a best response:*

   $v^{\lambda}(\boldsymbol{x}, j^{\lambda}(\boldsymbol{x})) \geq v^{\lambda}(\boldsymbol{x}, j), \forall 1 \leq \lambda \leq \Lambda, \forall 1 \leq j \leq J.$

3. *The follower breaks ties in favor of the leader:*

   $u^{\lambda}(\boldsymbol{x}, j^{\lambda}(\boldsymbol{x})) \geq u^{\lambda}(\boldsymbol{x}, j), \forall 1 \leq \lambda \leq \Lambda, \forall j$ *that is a best response to* $\boldsymbol{x}$ *as above*.

In general, finding the equilibrium of a Bayesian Stackelberg game is NP-hard [Conitzer and Sandholm, 2006].

2.1.3 Stackelberg Security Games

I now introduce a restricted version of the Stackelberg game knows as a security game. We consider a Stackelberg Security Games (SSG) [Yin et al., 2010] with a single leader (defender) and at least one follower (attacker). The defender has to protect a set of targets $T = \{t_1, t_2, ..., t_{|T|}\}$ from being attacked by the attacker, using a set of γ resources. In a security game, a pure strategy of an attacker is defined as attacking a single target; and a pure strategy of a defender is defined as an assignment of all the security resources to the set of targets. An assignment of a security resource to a target is also referred to as covering a target. The defender strategy set includes all the possible assignments of all the resources.

The payoffs for both the defender and the attacker depend on which target is attacked, and whether that target is protected (covered) by the defender. Formally, let d and a still denote the defender and the attacker respectively. We then use R_i^d to represent the defender's payoff (reward) of covering a target t_i that is attacked by the attacker, and P_i^d as the payoff (penalty) of not covering that attacked target. Similarly for the attacker, we use P_i^a (penalty) and R_i^a (reward) to represent his payoff of attacking a target t_i when it is covered or uncovered by the defender.

An important feature of the security game is that $R_i^d \geq P_i^d$, and that $P_i^a \leq R_i^a$. In other words, add resources to cover a target benefits the defender and hurts the attacker.

In many real world security problems, there are constraints on assigning the resources. For example, in the FAMS problem [Jain et al., 2010b], an air marshal is scheduled to protect 2 flights (targets) out of M total flights. The total number of possible schedule is $\binom{M}{2}$. However, not all of the schedules are feasible, since the flights scheduled for an air marshal have to be connected, e.g. an air marshal cannot be on a flight from A to B and then on a flight C to D. A resource assignment constraint implies that the feasible assignment set $\mathcal{A}$ is restricted; not all combinatorial assignment of resources to targets are allowed.

A compact representation of the defender strategy, introduced in [Kiekintveld et al., 2009], uses the probability that each target will be covered by a security resource. The defender's mixed-strategy can then be denoted by a vector $\boldsymbol{x} = \langle x_1, \ldots, x_{|T|} \rangle$, where c_i denote the probability that target t_i will be covered by a security resource. This compact representation of the defender strategies is proved to be equivalent to the distribution over the original pure strategies when there is no constraints on assigning the resources [Korzhyk et al., 2010]. In the presence of assignment constraints, such equivalence usually can be maintained by adding a set of linear constraints on $\boldsymbol{x}$ ($A\boldsymbol{x} \preceq b$).

Definition 2. *We consider a marginal coverage* **x** *to be feasible if and only if there exists* $a_j \geq 0, A_j \in \mathcal{A}$ *such that* $\sum_{A_j \in \mathcal{A}} a_j = 1$ *and for all* $i \in \mathcal{T}$, $x_i = \sum_{A_j \in \mathcal{A}} a_j A_{ij}$.

In fact, $\langle a_j \rangle$ is the mixed strategy over all the feasible assignments of the resources.

With this compact representation, efficient algorithms were able to be developed to compute defender optimal strategies [Kiekintveld et al., 2009; Tsai et al., 2010]. I will show more details

in Chapter 7 on the benefit of using this compact representation. Given the coverage vector $\boldsymbol{x}$, the defender's expected utility when the attacker attacks target t_i is calculated using Equation 2.1; and the attacker's expected utility of attacking target t_i is calculated in Equation 2.2.

$$U_i^d(\boldsymbol{x}) = (1 - x_i)P_i^d + x_i R_i^d \tag{2.1}$$

$$U_i^a(\boldsymbol{x}) = x_i P_i^a + (1 - x_i)R_i^a \tag{2.2}$$

2.2 Los Angeles International Airport

While there are a number of security problems where game theory is potentially applicable, I will focus on introducing the security scenario at the Los Angeles International Airport in this section. It is also the base of my experiment setup, due to its simplicity in constraints which is ideal for an initial investigation against human subjects. Los Angeles International Airport (LAX) is the fifth busiest airport in the United State, and the largest destination airport in the United State [Stevens et al., 2006]. It serves 60-70 million passengers each year [Stevens et al., 2006]. LAX is unfortunately one of the prime terrorist target on the west coast of the United State, given its importance and the record of the multiple attempting attacks by the arrested plotters [Stevens et al., 2006]. The Los Angeles World Airport (LAWA) police have designed a security system to protect the airport, which includes vehicular checkpoints, police units patrolling the roads to the terminals and inside the terminals (with canines), and security screening and bag checks for passengers. Unfortunately, there are not enough resources to protect the entire airport all the time, given the size of the airport and the number of passengers. Setting up available checkpoints, canine units or other patrols on deterministic schedules allows adversaries to learn the schedules

(a) LAX Checkpoint (b) Canine Patrol

Figure 2.1: LAX Security

and plot an attack that avoids the police checkpoints and patrols, which makes deterministic schedules ineffective.

Game-theoretic approach provides a solution to randomize the allocation of the limited resources for LAWA. In particular, the ARMOR [Pita et al., 2008] system is developed based on using the security game framework to assist LAWA. Figure 2.1(a) shows a vehicular checkpoint set up on a road inbound towards LAX. Police officers examine cars that drive by, and if any car appears suspicious, they do a more detailed inspection of that car. ARMOR provides a randomized schedule for the LAWA police to set up these checkpoints for a particular time frame. At the same time, ARMOR also generates an random assignment of canines to patrol routes through the terminals inside LAX. Figure 2.1(b) illustrates a canine unit on patrol at LAX.

2.3 Baseline Solvers

The leader's goal in a SSG is to maximize her expected utility, given how the adversary responds to the defender's strategy. The behavioral modeling is done only on the attacker, who faces a decision theory problem given the leader's commitment. Mathematically, the defender's optimal strategy can be computed by solving the following optimization problem:

$$\boldsymbol{x}^* = arg\max_{\boldsymbol{x}} \sum_i q_i(\boldsymbol{x}) U_i^d(\boldsymbol{x}) \tag{2.3}$$

where, $U_i^d(\boldsymbol{x})$ is the defender's expected utility if the attacker chooses to attack target t_i as shown in Equation 2.1, and $q_i(\boldsymbol{x})$ represents the attacker's response given defender's strategy $\boldsymbol{x}$.

One leading family of algorithms to compute such mixed strategies are DOBSS and its successors [Pita et al., 2008; Kiekintveld et al., 2009], which are used in the deployed ARMOR and IRIS applications. These algorithms follows the perfect rationality assumption for the adversary decision-making. However, in many real world domains, agents face human adversaries whose behavior may deviate from such assumption. COBRA [Pita et al., 2010] represents the best available benchmark for how to determine defender strategies in security games against human adversaries with $\epsilon-$optimal response. In this section, we describe the computation of the defender optimal strategies against two baseline models of the adversary: a perfectly rational adversary; and a $\epsilon-$optimal adversary response.

2.3.1 Defender Optimal Strategy against a perfectly rational adversary

The Strong Stackelberg Equilibrium assumes the adversary is perfectly rational, i.e. he will strictly maximizes his expected utility. The computation of the defender's optimal strategy can then be formulated as the following:

$$\max_{\boldsymbol{x},\boldsymbol{q}} \sum_i q_i U_i^d(\boldsymbol{x}) \tag{2.4}$$

$$\text{s.t.} \sum_{i=1}^{n} x_i \leq \Upsilon \tag{2.5}$$

$$0 \leq x_i \leq 1, \quad \forall i \tag{2.6}$$

$$q_i = 1, \text{if } U_i^a(\boldsymbol{x}) \geq U_{i'}^a(\boldsymbol{x}), \forall i' \neq i \tag{2.7}$$

$$\sum_i q_i = 1, \tag{2.8}$$

$$q_i \in \{0, 1\}, \forall i \tag{2.9}$$

The objective is to maximize the defender's expected utility, as shown in Equation (2.17). The constrains in Equations (2.7)-(2.9) enforce that the adversary selects the target which maximizes

his expected utility. By introducing some auxiliary variables, the above optimization problem can be formulated as a Mixed-Integer Linear Program (MILP), as shown below:

$$\max_{\boldsymbol{x},a,d,\boldsymbol{q}} \; d \tag{2.10}$$

$$\text{s.t.} \sum_{i=1}^{n} x_i \leq \Upsilon \tag{2.11}$$

$$0 \leq x_i \leq 1, \quad \forall i \tag{2.12}$$

$$0 \leq a - U_i^a(x_i) \leq M(1 - q_i), \forall i \tag{2.13}$$

$$\sum_i q_i = 1 \tag{2.14}$$

$$q_i \in \{0, 1\}, \forall i \tag{2.15}$$

$$M(1 - q_i) + U_i^d(x_i) \geq d, \forall i \tag{2.16}$$

The variable a in Equation 2.13 represents the attacker's expected utility. M is a very large constant, which enforces q_i to be set to 1 for the target that leads to the maximum expected utility for the attacker. Similarly, the variable d in the objective function and Equation (2.16) represents the defender's expected utility.

The defender's optimal strategy against a perfect rational adversary can then be computed by solving the above MILP.

2.3.2 Defender Optimal Strategy against the $\epsilon-$optimal adversary response

The $\epsilon-$optimal response addresses the bounded rationality of the adversary. It assumes that, instead of strictly maximizing the expected utility, the adversary could deviate to any target with

an expected utility within ϵ of the maximum. The computation of the defender's optimal strategy against $\epsilon-$optimal response can then be formulated as the following:

$$\max_{\boldsymbol{x},a,d,\boldsymbol{q},\boldsymbol{h}} \; d \tag{2.17}$$

$$\text{s.t.} \; \sum_{i=1}^{n} x_i \leq \Upsilon \tag{2.18}$$

$$0 \leq x_i \leq 1, \quad \forall i \tag{2.19}$$

$$0 \leq a - U_i^a(x_i) \leq M(1-q_i), \forall i \tag{2.20}$$

$$\sum_i q_i = 1 \tag{2.21}$$

$$q_i \in \{0,1\}, \forall i \tag{2.22}$$

$$\epsilon(1-h_i) \leq a - U_i^a(x_i) \leq M(1-q_i) + \epsilon, \forall i \tag{2.23}$$

$$h_i \in \{0,1\}, \forall i \tag{2.24}$$

$$q_i \leq h_i, \forall i \tag{2.25}$$

$$M(1-h_i) + U_i^d(x_i) \geq d, \forall i \tag{2.26}$$

Note that the above MILP modifies the MILP in Equation (2.17)-(2.16). The variable $\boldsymbol{h}$ in Equation (2.23)-(2.25) represents the $\epsilon-$optimal response of the adversary. h_i is set to 1 if the expected utility of attacking target t_i is within ϵ of a, which is the maximum expected utility the attacker can achieve.

2.4 Human Subject Experiments

Since my research is focused on addressing the boundedly rational behavior of human adversaries, conducting experiments with human subjects is necessary to evaluate the effectiveness

of the proposed approaches. To that end, I conduct my experiment with human subjects using an online labor market, i.e. Amazon Mechanical Turk (AMT). AMT has been widely used for behavioral research as a tool to collect data [Mason and Suri, 2012]. The are many advantages of conducting experiment on AMT, including subject pool access, subject pool diversity and low cost [Reips, 2002; Mason and Suri, 2012]. While conducting experiments with real terrorists is often infeasible in reality, experimental analysis with general population still points to the right direction and allows me to show how my approaches are expected to perform compared to alternative approaches.

One might argue that the psychiatric profile of the terrorists might significantly differ from the general population. Therefore, the terrorists are completely irrationally and not making any strategic decisions in planning the attack. However, studies show that the normalcy is indeed the primary shared characteristic of the psychiatric profile of the terrorists [Richardson, 2007; Abrahms, 2008; Gill and Young, 2011]. In fact, they are highly rational and carefully conducting the attacking plan [Richardson, 2007; Rosoff and John, 2009; Keeney and von Winterfeldt, 2010].

It then follows the question whether the perfect rationality assumption is sufficient for modeling the decision-making of the terrorists. First of all, many studies in economic behavior and cognitive science show that human decision makers suffer from bounded rationality and cognitive limitation. The bounded rationality of human decision makers may be caused by both external and internal reasons[Simon, 1956, 1969; Hastie and Dawes, 2001]. On the one hand, the environment may be complicated. The human decision makers might only have limited information of the environment. On the other hand, humans have limited memory and other cognitive limitation which prevent them from making optimal choice. Indeed, many studies in the literature [Rubinstein, 1998; Camerer, 2003] have shown that human decision makers rely on heuristics in making

decisions rather than strictly maximize the expected utility. Furthermore, terrorists sometimes face competing objectives and noisy information [Allison and Zelikow, 1999; Abrahms, 2008], which may lead them to deviate from the optimal strategy.

Additionally, the approaches developed in this thesis based on using human-subject experiments with general populations may be of use beyond the counter-terrorism domain. In many other domains, the criminals are more close to the general population, such as the ticket-less travelers in the metro train system, or the villagers illegally hunting animals or extracting plants. In general, criminal activities can be broadly broken down into six categories: (i) Property crimes, (ii) violent crimes, (iii) sex crimes, (iv) gangs and crime, (v) white-collar occupational crime, and (vi) drugs and crime [Pogrebin, 2012]. The responsibility of different security agencies is to prevent these crime activities. The approaches presented in this thesis can be potentially helpful to many of these agencies. Given the large range of crime activities, the human criminals will also span a wide variety. Therefore, the use of general population in the human subject experiment can be of great value for providing insights of how the proposed approach might be applicable to these different domains.

In the future, we could further refine the approach for a specific type of criminal. More specifically, by defining the personality and demographic profile of the criminals in a specific type of crimes, we can evaluate the approach in experimental with human subjects of that profile. However, the difficulty of obtain the correct population that needs to be examined is general in behavioral studies and might not be complete tackled.

Chapter 3: Related Work

Motivated by real-world security problems, there have been many algorithms developed to compute optimal defender strategies in Stackelberg games [Paruchuri et al., 2008; Kiekintveld et al., 2009; Tsai et al., 2010]. The first such algorithm to be used in a real application is DOBSS (Decomposed Optimal Bayesian Stackelberg Solver) [Paruchuri et al., 2008], which is the central to the ARMOR system [Pita et al., 2008] at LAX airport and the GUARDS system [Pita et al., 2011] built for the Transportation Security Administration. Other works related to Stackelberg security games include those of Agmon et al. [Agmon et al., 2008, 2009] and those of Gatti et al. [Gatti, 2008b; Basiloco et al., 2009] on multi-robot patrolling. However, an important limitation of all of this work is the assumption of a perfectly rational adversary, which may not hold in many real world domains.

3.1 Behavioral Game Theory

Behavioral Game Theory aims at developing models of human decision-making in strategic settings. Many models have been proposed to capture human bounded rationality in their decision making in the literature of psychology and cognitive science [Train, 2003; McFadden, 1989; Starmer, 2000; Rubinstein, 1998]. A key challenge of applying these models to game-theoretical

framework to help design better strategy is the transition from a (sometimes descriptive) model to a computational model. On the other hand, there has been a growing interests in the game theory literature in developing more realistic computational models incorporating human decision making in games [Camerer et al., 2004; Ficici and Pfeffer, 2008; Stahl and Wilson, 1994]. Most of these models find empirical support from the data of human playing games. However, few research efforts have being made to identify which of these models capture the salient features of human decision-making in the important area of SSGs. To that end, my work focus on extending the existing models from literature to apply to SSGs as well as designing experiments to evaluate the effectiveness of these models with human subjects.

The most related work to this thesis is that by Pita et al.[Pita et al., 2010]. Pita et al. develop a new algorithm COBRA, which provids a solution for designing better defender strategies against human adversaries by considering two factors in human behavior (i) human deviation from the utility maximizing strategy and (ii) human anchoring bias when given limited observation of defender mixed strategy. COBRA significantly outperforms the baseline algorithm DOBSS, which assumes perfect rationality of the adversaries, in the experiments against human subjects, and is considered the leading contender in addressing human bounded rationality in SSG. However, COBRA only exploits two aspects for human bounded rationality. There are many other models proposed in the literature of behavior game theory and cognitive psychology which could be potentially used to model adversary decision-making in Stackelberg security games. Thus, it remains an open question whether there are other approaches that allow for fast solutions and outperform COBRA in addressing human behavior in security games.

Outside the area of Stackelberg security games, there have been several recent investigations of human subjects interacting with agents. For example, Melo et al [de Melo et al., 2011] investigate the impact of expression of an automated agent's anger or happiness in how a human participant may play the game. In repeated prisoner's dilemma games, agents' expressions are shown to significantly affect human subjects' cooperation or defection. Similarly, Azaria et al. [Azaria et al., 2011] focus on road selection games, and advice an automated system may provide to human subjects; Peled et al. [Peled et al., 2011] focus on bilateral bargaining games, designing agents that negotiate proficiently with people. Aside from the obvious difference that our focus is on SSGs, another key is our focus on efficiently computing optimal mixed strategies for the defender.

3.2 Efficient Computation of Defender Optimal Strategy

There have been a number of algorithms developed to compute the optimal defender strategy for massive real-world security problems [Paruchuri et al., 2008; Kiekintveld et al., 2009; Jain et al., 2010a]. In order to scale-up the computation for SSG with combinatorial number of defender strategies, Kiekintveld et al. [Kiekintveld et al., 2009] exploit the underlying structure of security games and developed efficient algorithms based on using a compact representation of the defender strategy. More specifically, ORIGAMI computes the optimal defender strategy in polynomial time when there is no constraints on assigning the security resources; ERASER-C provides scales-up over ORIGAMI by computing the security coverage per schedule instead of computing the mixed-strategy over a joint assignment for all security resources. However, Kiekintveld et al. [Kiekintveld et al., 2009] show that ERASER-C only addresses certain types of

constraints on assigning the resources [Korzhyk et al., 2010] and may fail to produce a correct solution when facing arbitrary constraints. Jain et al. [Jain et al., 2010a] then develop a novel algorithm ASPEN based on using the branch-and-price approach. ASPEN further advanced the state of art and is able to handle arbitrary resource allocation constraint. In essential, ASPEN avoids representing the full space of defender pure strategies by starting with a small subset of it and iteratively expanding it until reaches the optimal solution.

At the same time, many algorithms have been developed to solve Bayesian Stackelberg games. DOBSS [Paruchuri et al., 2008] is the first algorithm develop for computing defender optimal strategy in a Bayesian Stackelberg game. Jain [Jain et al., 2011b] then proposed a hierarchical methodology of decomposing large Bayesian Stackelberg games into many smaller Bayesian Stackelberg games, and provided a framework to use the solutions to these smaller games to efficiently apply branch-and-bound on the original large Bayesian Stackelberg game. Yin et al. [Yin and Tambe, 2012] further improved the state-of-art by combining techniques in artificial intelligence such as best-first search and operation research such as Bender's decomposition.

Unfortunately, all the previous work assumes a perfectly rational adversary. Given that most of the behavioral models imply mathematically complex presentations of the adversary decision making, it is unclear whether similar techniques can be applied for computing defender optimal strategy incorporating these models. Therefore, new efficient algorithms need to be developed to address this new computational challenge.

3.3 Robust Defender Strategies

Another line of related work in Stackelberg security games has been trying to design more robust strategies to deal with different kinds of uncertainties [Aghassi and Bertsimas, 2006; Yin et al., 2011; Kiekintveld et al., 2011]. Yin et al. [Yin et al., 2011] proposed a unified efficient algorithm that addresses both execution uncertainties of the defender and observation uncertainties of adversaries in SSGs. Kiekintveld et al. [Kiekintveld et al., 2011] address payoff uncertainty by introducing a general model of infinite Bayesian Stackelberg security games which allows payoffs to be represented using continuous payoff distributions. An et al. [An et al., 2012] considers the cases where the adversaries don't have perfect surveillance of the defender's strategy. They provided a model for the adversary's belief update of the defender's strategy as well as the formulation for computing the defender's optimal strategy considering such imperfect surveillance of the adversary. A key difference of this line of work from this thesis is that this work considers robustness against perfectly rational adversaries. Although the simulation based experiment showed promising result of these studies, the performances of these models against real human subjects are left unaddressed.

In order to address adversary bounded rationality, Pita et al. [Pita et al., 2012] introduce a new algorithm MATCH which computes a robust strategy for the defender with a linear correlated cost of the defender to that of the adversary. More specifically, MATCH guarantees that if the adversary deviates from his optimal action, the cost of such deviation to the defender is linearly correlated to that to the adversary. MATCH is designed to intentionally avoid explicitly modeling the decision making of the human adversary. It is unclear how such approach compares to that based on building an explicit model to predict the adversary's decision making.

3.4 Learning Adversary Behavioral in Repeated Games

There is a significant body of literature in game theory on learning with incomplete information [Brown, 1951; Sastry et al., 1994; Aumann and Maschler, 1995]. These studies focus on addressing the uncertainty of payoff information of the game in repeated game settings via learning. The players are often assumed to be perfectly rational in these studies. In the scope of security games, there has been work on learning attacker payoffs in repeated security games [Letchford et al., 2009; Marecki et al., 2012]. Letchford et al. [Letchford et al., 2009] develop an algorithm to uncover the attacker type in as few rounds as possible. Marecki et al. [Marecki et al., 2012] use Monte-Carlo Tree Search to maximize the defenders utility in the first few rounds. Either work provides a guidance for the defender in the later round.

In a most recent work [Qian et al., 2014], Qian et al propose an algorithm combining Gibbs sampling with Monte Carlo tree search for online planning for the defender. In comparison, the work presented in this thesis focuses on learning the behavioral model of the adversaries from past crime data, and also providing guidance for defender to adapt their resource allocation strategy based on the updated belief of the model. Furthermore, this thesis addresses the interesting problem of learning the behavioral model from both labeled and unlabeled crime data, similar to that concerned by semi-supervised learning problems[Chapelle et al., 2006].

Chapter 4: Modeling Adversary Decision Making

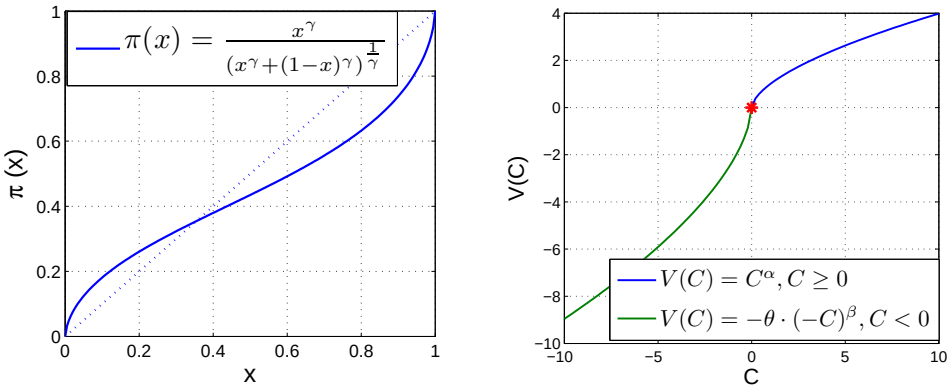
This chapter introduces my contribution towards moving beyond perfect rationality assumptions of human adversaries in security games. In order to integrate more realistic models of human decision-making in real-world security systems, several key challenges need to be addressed. First, the literature has introduced a multitude of potential models on human decision making [Kahneman and Tvesky, 1979; Camerer et al., 2004; McKelvey and Palfrey, 1995; Costa-Gomes et al., 2001], but each of these models has its own set of assumptions and there is little consensus on which model is best for different types of domains. Therefore, there is an important empirical question of which model best represents the salient features of human behavior in the important class of applied security games. Second, integrating any of the proposed models into a decision-support system (even for the purpose of empirically evaluating the model) requires developing new algorithms for computing solutions to Stackelberg security games, since most existing algorithms are based on mathematically optimal attackers [Paruchuri et al., 2008; Kiekintveld et al., 2009]. One notable exception is COBRA developed by Pita et al. [Pita et al., 2010]. COBRA is one example of modeling bounded rationality of human adversaries by taking into account (i) the anchoring bias of humans while interpreting the probabilities of several events; (ii) the limited computational ability of humans which may lead to deviation from their best response. To the

best of our knowledge, COBRA is the best performing strategy for Stackelberg security games in experiments with human subjects. Thus, the open question is whether there are other approaches that allow for fast solutions and outperform COBRA in addressing human behavior in security games.

This chapter significantly expands the previous work on modeling human behavior in Stackelberg security games. Section 4.1 presents the new models of adversary decision-making based on Prospect Theory and Quantal Response. Following that, Section 4.2 describes the algorithms we developed to compute optimal defender strategy against these new adversary models. In Section 4.3, we explain the methods we used to decide the parameters of different models. Section 4.4 presents our experimental setup and results.

4.1 Models for Predicting Attacker Behaviors

Existing models of adversary behavior in SSGs have poor performance in predicting the behavior of human adversaries [Pita et al., 2010]. In order to design better defender strategy, better models of adversary decision-making need to be developed. In this section, we present three models of adversary's behavior in SSGs, based on using Prospect Theory and Quantal Response Equilibrium. All of the models have key parameters. We describe in the next section our methodology for setting these parameters in each case.



(a) weighting function (b) value function

Figure 4.1: Prospect Theory empirical function forms

4.1.1 Prospect Theory

Prospect Theory provides a descriptive model of how humans make decision among alternatives with risk, which is a process of maximizing the 'prospect', which will be defined soon, rather than the expected utility. More formally, the prospect of a certain alternative is defined as

$$\sum_l \pi(x_l)V(C_l) \tag{4.1}$$

In Equation (4.1), x_l denotes the probability of receiving C_l as the outcome. The weighting function $\pi(\cdot)$ describes how probability x_l is perceived by individuals. An empirical function form of $\pi(\cdot)$ (Equation (4.2)) is shown in Fig. 4.1(a) [Kahneman and Tvesky, 1992].

$$\pi(x) = \frac{x^\gamma}{(x^\gamma + (1-x)^\gamma)^{\frac{1}{\gamma}}} \tag{4.2}$$

The key concepts of a weighting function are that individuals overestimate low probability and underestimate high probability [Kahneman and Tvesky, 1979, 1992]. Also, $\pi()$ is not consistent with the definition of probability, i.e. $\pi(x) + \pi(1-x) \leq 1$ in general.

The value function $V(C_l)$ in Equation (4.1) reflects the value of the outcome C_l. PT predicts that individuals are risk averse regarding gain but risk seeking regarding loss, implying an

S-shaped value function [Kahneman and Tvesky, 1979, 1992]. A key component of Prospect Theory is the reference point. Outcomes lower than the reference point are considered as loss and higher as gain.

$$V(C) = \begin{cases} C^{\alpha}, & C \geq 0 \\ -\theta(-C)^{\beta}, & C < 0 \end{cases} \tag{4.3}$$

Equation (4.3) is a general form for the value function where C is the relative outcome to the reference. In Equation (4.3), we assume the reference point to be at 0. α and β determine the extent of non-linearity in the curves. If the parameters $\alpha = 1.0$ and $\beta = 1.0$, the function would be linear; typical values for both α and β are 0.88 [Kahneman and Tvesky, 1992]. θ captures the idea that the loss curve is usually steeper than the gains curve, a typical value of θ is 2.25 [Kahneman and Tvesky, 1992], which reflects a finding that losses are a little more than twice as painful as gains are pleasurable. The function is also displayed in Fig. 4.1(b) [Kahneman and Tvesky, 1992]. Given these parameters, we will henceforth denote this value function with $V_{\alpha,\beta,\theta}$

In a SSG, the prospect of attacking target t_i for the adversary is computed as

$$\texttt{prospect}(t_i) = \pi(x_i)V_{\alpha,\beta,\theta}(P_i^a) + \pi(1 - x_i)V_{\alpha,\beta,\theta}(R_i^a) \tag{4.4}$$

According to Prospect Theory, subjects will choose the target with the highest prospect. Thus,

$$q_i = \begin{cases} 1, \text{ if } \texttt{prospect}(t_i) \geq \texttt{prospect}(t_{i'}), \forall t_{i'} \in T \\ 0, \text{ otherwise} \end{cases} \tag{4.5}$$

4.1.2 Quantal Response

Quantal Response is an important solution concept in behavioral game theory [McKelvey and Palfrey, 1995]. It is based on a long history of work in single-agent problems and brings that work into a game-theoretic setting [Stahl and Wilson, 1994; Wright and Leyton-Brown, 2010]. It

assumes that instead of strictly maximizing utility, individuals respond stochastically in games: the chance of selecting a non-optimal strategy increases as the cost of such an error decreases. Given the strategy profile of all the other players, the response of a player is modeled as a quantal response (QR model): he/she selects action i with a probability given by

$$q_i(x) = \frac{e^{\lambda U_i^a(x)}}{\sum_{t_k \in T} e^{\lambda U_k^a(x)}} \tag{4.6}$$

where, $U_i^a(x)$ is the expected utility for the attacker for selecting pure strategy i. Here, $\lambda \in [0, \infty]$ is the parameter that captures the rational level of player p: one extreme case is λ=0, when player p plays uniformly random; the other extreme case is $\lambda \to \infty$, when the quantal response is identical to the best response. Combining Equation(4.6) and (2.2),

$$q_i(x) = \frac{e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)x_i}}{\sum_{t_k \in T} e^{\lambda R_k^a} e^{-\lambda(R_k^a - P_k^a)x_k}} \tag{4.7}$$

In applying the QR model to the security game domain, we only consider noise in the response of the adversary. The defender uses a computer decision support system to choose her strategy hence is able to compute optimal strategy. On the other hand, since the attacker observes the defender's strategy first to decides his response, it can only hurt the defender to add noise in her response. Recent work [Wright and Leyton-Brown, 2010] shows Quantal Level-k [Stahl and Wilson, 1994] to be best suited for predicting human behavior in simultaneous move games. The key idea of level-k is that humans can perform only a bounded number of iterations of strategic reasoning: a level-0 player plays randomly, a level-k ($k > 1$) player best response to the level-$(k-1)$ player. We applied QR instead of Quantal Level-k to model the attacker's response because in Stackelberg security games the attacker observes the defender's strategy, so level-k reasoning is not applicable.

4.1.3 Quantal Response with Rank-related Expected Utility

We modify the Quantal Response Model by taking into consideration the fact that individuals are attracted to extreme events, such as the less uncertain and highest payoff. This idea is inspired by the rank-dependent Expected Utility Model [Diecidue and Wakker, 2001], in which the utilities of choosing different alternatives are based on the their ranks. We adapt this idea to security games, but we only consider such effect on the target covered with minimum resources. That is the adversary would prefer the target covered with minimum resources since he is most likely to be successful attacking that target. This could significantly reduce the defender's reward in the case when this target with fewest resources also gives a large penalty to the defender.

We modify the QR model by adding extra weight to the target covered with minimum resources. We refer this modified model as Quantal Response with Rank-related expected Utility (QRRU) model, where the probability that the attacker attacks target t_i is computed as

$$q_i(x) = \frac{e^{\lambda_u U_i^a(x_i)} e^{\lambda_s S_i(x)}}{\sum_{t_k \in T} e^{\lambda_u U_k^a(x_k)} e^{\lambda_s S_k(x)}} \tag{4.8}$$

where $S_i(x) \in \{0, 1\}$ indicating whether t_i is covered with least resource.

$$S_i(x) = \begin{cases} 1, \text{ if } x_i \leq x_i', \forall t_{i'} \in T \\ 0, \text{ otherwise} \end{cases} \tag{4.9}$$

The denominator in Equation (4.8) is only for normalizing the probability distribution so all the q_i sum up to 1. In the numerator, we have two terms deciding the probability that target t_i will be chosen by the adversary. The first term $e^{\lambda_u U_i^a(x_i)}$ relates to the expected utility for the adversary to choose target t_i. $U_i^a(x_i)$ is computed as in Equation (2.2). The parameter $\lambda_u \geq 0$ represents the level of error in adversary's computation of the expected utility, which is equivalent to λ in Equation (4.6). The second term $e^{\lambda_s S_i(x)}$ relates to the adversary's preference for the least

covered target. Note that if t_i is not covered with the minimum resource, this term equals to 1 so there is no extra weight added to the non-minimum covered targets; if t_i is covered with minimum resource, this term will be ≥ 1, adding extra weight to the probability that adversary will choose t_i. The parameter $\lambda_s \geq 0$ represents the level of the adversary's preference to the minimum covered target. $\lambda_s = 0$ indicates no preference to the minimum covered target. As λ_s increase, this preference becomes stronger.

4.2 Computing Optimal Defender Strategy

Given the new models of adversary behavior in SSG, new algorithms need to be developed to compute the optimal defender strategy since the existing algorithms are based on the assumption of a perfectly rational adversary. We now describe efficient computation of the optimal defender mixed strategy assuming a human adversary whose response follows one of the three models we proposed: Prospect Theory (PT-Adversary), Quantal Response (QR-Adversary) or Quantal Response with Rank-related Utility (QRRU-Adversary).

4.2.1 Computing against a PT-adversary

Assuming that the adversary's response follows Prospect Theory (PT-adversary), we developed two methods to compute the optimal defender strategy.

4.2.1.1 BRPT

Best Response to Prospect Theory (BRPT) is a mixed integer programming formulation for computing the optimal leader strategy against players whose responses follow a PT model. We first present an abstract version of our formulation of BRPT in Equations (4.10)-(4.14), and then

present a more detailed operational version in Equations (4.15)-(4.27) that uses piecewise linear approximation to provide the BRPT MILP (Mixed Integer Linear Program).

$$\max_{x,q,a,d,z} \; d \tag{4.10}$$

$$\text{s.t.} \; \sum_{i=1}^{n} x_i \leq M \tag{4.11}$$

$$\sum_{i=1}^{n} q_i = 1, \; q_i \in \{0,1\} \tag{4.12}$$

$$0 \leq a - (\pi(x_i)V(P_i^a) + \pi(1-x_i)V(R_i^a)) \leq K(1-q_i), \forall i \tag{4.13}$$

$$K(1-q_i) + (x_i R_i^d + (1-x_i)P_i^d) \geq d, \forall i \tag{4.14}$$

The objective is to maximize d, the defender's expected utility. Equation (4.11) enforces that the constraint on the total amount of resources is met. In Equation (4.12), the integer variables q_i represent the attacker's pure strategy. In BRPT, q_i is constrained to be binary variable, since, as justified and explained in [Paruchuri et al., 2008], we assume the adversary has a pure strategy best response: $q_i = 1$ if t_i is attacked and 0 otherwise. Equation (4.13) is the key to decide the attacker's strategy, given a defender's mixed strategy $x =< x_i >$. The variable a represents the attacker's 'benefit' of choosing a pure strategy $< q_i >$. Since we are modeling attacker's decision making using Prospect Theory, the benefit perceived by the adversary for attacking target t_i is the attacker's 'prospect', which is calculated as $(\pi(x_i)V(P_i^a)+\pi(1-x_i)V(R_i^a))$ following Equation (4.1). The attacker tries to maximize a by choosing the target with the highest 'prospect', as enforced by Equation (4.13). In particular, the inequality on the left side of Equation (4.13) enforces that a is greater or equal to the 'prospect' of attacking any target. On the right hand of Equation (4.13), we have a constant parameter K with a very large positive value. For targets

with $q_i = 0$, the upper bound of the difference between a and the 'prospect' is K, therefore, the bounds is not operational. For target with $q_i = 1$ (i.e. the target chosen by the attacker), the value of a is forced to be equal to the actual 'prospect' of attacking that target. In Equation (4.14), the constant parameter K enforces that d is only constrained by the target that is attacked by the adversary (i.e. $q_i = 1$).

We now present the BRPT MILP based on our piecewise linear approximation of the weighting function as discussed earlier. We use the empirical functions introduced in Section 4.1.1 for the weighting function $\pi(\cdot)$ and value function $V(\cdot)$. Let $(P_i^a)' = V(P_i^a)$ and $(R_i^a)' = V(R_i^a)$ denote the adversary's value of penalty P_i^a and reward R_i^a, which are both given as input to the optimization formula in Equations (4.11)-(4.14). The key challenge to solve that optimization problem is that the $\pi(\cdot)$ function is non-linear and non-convex. If we apply the function directly, we have to solve a nonlinear and non-convex mixed-integer optimization problem, which is difficult. Therefore, we approximately solve the problem by representing the non-linear $\pi(\cdot)$ function as a piecewise linear function. This transforms the problem into a MILP, which is shown in Equations (4.15)-(4.27).

$$\max_{x,q,a,d,z} \; d \tag{4.15}$$

$$\text{s.t.} \sum_{i=1}^{n}\sum_{k=1}^{5} x_{ik} \leq M \tag{4.16}$$

$$\sum_{k=1}^{5}(x_{ik} + \bar{x}_{ik}) = 1, \forall i \tag{4.17}$$

$$0 \leq x_{ik}, \bar{x}_{ik} \leq c_k - c_{k-1}, \forall i, k = 1..5 \tag{4.18}$$

$$z_{ik} \cdot (c_k - c_{k-1}) \leq x_{ik}, \forall i, k = 1..4 \tag{4.19}$$

$$\bar{z}_{ik} \cdot (c_k - c_{k-1}) \leq \bar{x}_{ik}, \forall i, k = 1..4 \tag{4.20}$$

$$x_{i(k+1)} \leq z_{ik}, \forall i, k = 1..4 \tag{4.21}$$

$$\bar{x}_{i(k+1)} \leq \bar{z}_{ik}, \forall i, k = 1..4 \tag{4.22}$$

$$z_{ik}, \bar{z}_{ik} \in \{0, 1\}, \forall i, k = 1..4 \tag{4.23}$$

$$x'_i = \sum_{k=1}^{5} b_k x_{ik}, \; \bar{x}'_i = \sum_{k=1}^{5} b_k \bar{x}_{ik}, \forall i \tag{4.24}$$

$$\sum_{i=1}^{n} q_i = 1, \; q_i \in \{0, 1\} \tag{4.25}$$

$$0 \leq a - (x'_i(P^a_i)' + \bar{x}'_i(R^a_i)') \leq M(1 - q_i), \forall i \tag{4.26}$$

$$M(1 - q_i) + \sum_{k=1}^{5}(x_{ik}R^d_i + \bar{x}_{ik}P^d_i) \geq d, \forall i \tag{4.27}$$

Let $\tilde{\pi}(\cdot)$ denote the use of a piecewise linear approximation of the weighting function $\pi(\cdot)$, as shown in Figure 4.2. We empirically set 5 segments[1] for $\tilde{\pi}(\cdot)$. This function is defined by $\{c_k | c_0 = 0, c_5 = 1, c_k < c_{k+1}, k = 0, ..., 5\}$ that represent the endpoints of the linear segments and $\{b_k | k = 1, \ldots, 5\}$ that represent the slope of each linear segment. In order to represent the

[1]This piecewise linear representation of $\pi(\cdot)$ achieves a small approximation error: $\sup_{z\in[0,1]} \|\pi(z) - \tilde{\pi}(z)\| \leq 0.03$.

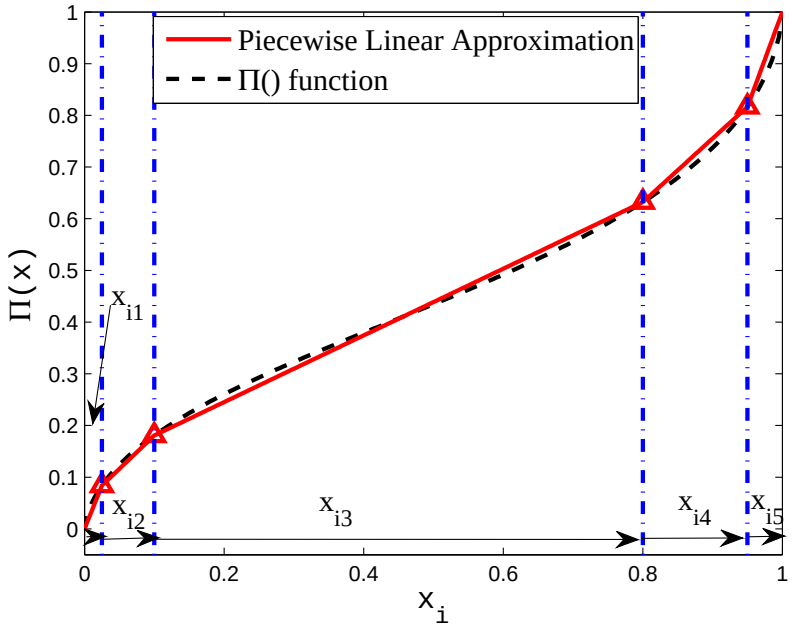


Figure 4.2: Piecewise approximation of the weighting function

piecewise linear approximation, i.e. $\tilde{\pi}(x_i)$ (and simultaneously $\tilde{\pi}(1-x_i)$), we partition x_i (and $1-x_i$) into five segments, denoted by variables x_{ik} (and $\bar{x}_{ik}$). Therefore, x'_i which equals $\tilde{\pi}(x_i)$ can be calculated as the sum of the linear function in each segment

$$x'_i = \tilde{\pi}(x_i) = \sum_{k=1}^{5} b_k \cdot x_{ik}$$

which is shown in Equation (4.24). At the same time, we can enforce the correctness of partitioning x_i (and $1-x_i$) by ensuring that segment x_{ik} (and $\bar{x}_{ik}$) is positive only if the previous segment is used completely. This is enforced in Equations (4.17)$\sim$(4.23) by using the auxiliary integer variable z_{ik} (and $\bar{z}_{ik}$). $z_{ik} = 0$ indicates that the k^{th} segment of x_i (i.e. x_{ik}) has not been completely used, therefore, the following segments can only be set to 0, and vice versa. Equation (4.24) defines x'_i=$\tilde{\pi}(x_i)$ as the value of the piecewise linear approximation of x_i, and $\bar{x}'_i$=$\tilde{\pi}(1-x_i)$ as the value of the piecewise linear approximation of $1-x_i$.

4.2.1.2 RPT

Robust-PT (RPT) modifies the base BRPT method to account for the possible uncertainty in adversary's choice caused (for example) by imprecise computations [Simon, 1956]. Similar to COBRA, RPT assumes that the adversary may choose any strategy within ϵ of the best choice, defined here by the prospect of each action. It optimizes the worst-case outcome for the defender among the set of strategies that have the prospect for the attacker within ϵ of the optimal prospect.

$$\max_{x,h,q,a,d,z} d \tag{4.28}$$

s.t. Constraints (4.16)$\sim$(4.26)

$$\sum_{i=1}^{n} h_i \geq 1 \tag{4.29}$$

$$h_i \in \{0,1\},\ q_i \leq h_i, \forall i \tag{4.30}$$

$$\epsilon(1-h_i) \leq a - (x_i'(P_i^a)' + \bar{x}_i'(R_i^a)') \leq M(1-h_i) + \epsilon, \forall i \tag{4.31}$$

$$M(1-h_i) + \sum_{k=1}^{5}(x_{ik}R_i^d + \bar{x}_{ik}P_i^d) \geq d, \forall i \tag{4.32}$$

We modify the BRPT optimization problem as follows: the first 11 constraints are equivalent to those in BRPT (Equation (4.16)-(4.26)); in Equation (4.29), the binary variable h_i indicates the ϵ-optimal strategy for the adversary; the ϵ-optimal assumption is embedded in Equation (4.31), which forces $h_i = 1$ for any target t_i that leads to a prospect within ϵ of the optimal prospect, i.e. a; Equation (4.32) enforces d to be the minimum expected utility for defender on the targets that lead to ϵ-optimal prospect for the attacker. RPT attempts to maximize the minimum for the defender over the ϵ-optimal targets for the attacker, thus providing robustness against attacker (human) deviations within that ϵ-optimal set of targets.

4.2.2 Computing against a QR-adversary

Assuming the adversary follows a quantal response (QR-adversary), we now present the algorithm to compute the defender's optimal strategy against a QR-adversary. Given the quantal response of the adversary, which is described in Equation (4.7), the best response of defender is to maximize her expected utility:

$$\max_x \quad U^d(x) = \sum_{i=1}^{n} q_i(x) U_i^d(x)$$

Combined with Equation (4.7) and (2.1), the problem of finding the optimal mixed strategy for the defender can be formulated as

$$\max_x \frac{\sum_{t_i \in T} e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)x_i}((R_i^d - P_i^d)x_i + P_i^d)}{\sum_{t_k \in T} e^{\lambda R_k^a} e^{-\lambda(R_k^a - P_k^a)x_k}} \tag{4.33}$$

$$\text{s.t.} \ \sum_{i=1}^{n} x_i \leq M \tag{4.34}$$

$$0 \leq x_i \leq 1, \quad \forall i, j \tag{4.35}$$

Algorithm 1: BRQR

```
opt_g ← −∞;
for it ← 1, ..., IterN do
    x^(0) ← randomly generate a feasible starting point;
    (opt_l, x*) ← Find-Local-Minimum(x^(0));
    if opt_g > opt_l then
        opt_g ← opt_l, x^opt ← x*;
    end
end
return opt_g, x^opt;
```

Unfortunately, since the objective function in Equation (4.33) is non-linear and non-convex, finding the global optimum is extremely difficult. Therefore, we focus on methods to find local optima. To compute an approximately optimal strategy against a QR-adversary efficiently, we

develop the Best Response to Quantal Response (BRQR) heuristic described in Algorithm 1. We first take the negative of Equation (4.33), converting the maximization problem to a minimization problem. In each iteration, we find the local minimum using the *fmincon*() function in Matlab with the Interior Point Algorithm with a given starting point. If there are multiple local minima, by randomly setting the starting point in each iteration, the algorithm will reach different local minima with a non-zero probability. By increasing the iteration number, $IterN$, the probability of reaching the global minimum increases. We empirically set $IterN$ to 300 in our experiments.

4.2.3 Computing against a QRRU-adversary

We now present the algorithm to compute defender optimal strategy assuming the adversary's behavior follows the QRRU model. The adversary's response given this model is computed as in Equation (4.8). The optimal defender strategy against a QRRU-adversary is computed by solving the following optimization problem:

$$\max_{x,s,x_{min}} \frac{\sum_{t_i \in T} e^{\lambda_u R_i^a} e^{-\lambda_u (R_i^a - P_i^a) x_i} e^{\lambda_s s_i} ((R_i^d - P_i^d) x_i + P_i^d)}{\sum_{t_k \in T} e^{\lambda_u R_k^a} e^{-\lambda_u (R_k^a - P_k^a) x_k} e^{\lambda_s s_k}} \tag{4.36}$$

s.t. Constraint $(4.34), (4.35)$

$$x_i - (1 - s_i)K \leq x_{min} \leq x_i, \forall t_i \in T \tag{4.37}$$

$$\sum_{t_i \in T} s_i = 1 \tag{4.38}$$

$$s_i \in \{0, 1\}, \forall t_i \in T \tag{4.39}$$

where the integer variables s_i are introduced to represent the function $S_i(x)$ as shown in Equation (4.9). In constraint (4.37), K is a constant with a very large value. Constraints (4.37) and (4.38) enforces x_{min} to be the minimum value among all the x_i. Simultaneously, s_i is set to 1 if target t_i

has the minimum coverage probability assigned; and is set to 0 otherwise. The above optimization problem is a non-linear and non-convex mixed integer programming problem, which is difficult to solve directly. Therefore, we developed Best Response to a QRRU-Adversary (BRQRRU), an algorithm that iteratively computes the defender's optimal strategy. The iterative approach breaks down the mixed-integer non-linear programming problem into sub-problems without integer variables. For each sub-problem, one of the target is assumed to be the least covered target. Then, under this constraint, the maximum defender expected utility and the associated defender mixed strategy are computed by solving a non-linear programming problem (similar to BRQR). Finally, the sub-problem generating the highest maximum defender expected utility is found as the 'actual' optimal solution, and the associated defender mixed-strategy is the optimal defender strategy assuming a QRRU-adversary.

Algorithm 2 shows the pseudo code of the algorithm. Algorithm 2 describes BRQRRU. In

Algorithm 2: BRQRRU

```
1 opt_g ← −∞;
2 for t_i' ∈ T do
3 |   (opt_l, x*) ← Find-Optimal-Defender-Strategy(s_i' = 1);
4 |   if opt_g > opt_l then
5 |   |   opt_g ← opt_l, x^opt ← x*;
6 |   end
7 end
8 Return opt_g, x^opt;
```

each iteration, one target $t_{i'}$ is conditioned to be covered with minimum resource, therefore $s_{i^*} = 1$. This reduces the optimization problem to the following

$$\max_{x} \frac{\sum_{t_i \in T} e^{\lambda_u R_i^a} e^{-\lambda_u (R_i^a - P_i^a) x_i} e^{\lambda_s s_i} ((R_i^d - P_i^d) x_i + P_i^d)}{\sum_{t_k \in T} e^{\lambda_u R_k^a} e^{-\lambda_u (R_k^a - P_k^a) x_k} e^{\lambda_s s_k}} \tag{4.40}$$

s.t. Constraint $(4.34), (4.35)$

$$x_{i^*} \leq x_i, \quad \forall t_i \in T \tag{4.41}$$

where there are no integer variables involved since $s_i, \forall t_i \in T$ are all pre-defined parameters of the optimization problem. Therefore, we could solve it using the same method of local search with random restart as that in BRQR. `Find-Optimal-Defender-Strategy`$(s_{i'} = 1)$ on Line (3) in Algorithm 2 calls Algorithm 1 to solve the optimization problem in Equation (4.40)-(4.41).

4.3 Parameter Estimation

In this section, we describe our methodology for setting the values of the parameters for the different models of human behavior introduced in the previous section. We set the parameters for our later experiments using data collected in a preliminary set of experiments with human subjects playing the online game which will be introduced in Section 4.4.1. We posted the game on Amazon Mechanical Turk as a Human Intelligent Task (HIT) and asked subjects to play the game. Subjects played the role of the adversary and were able to observe the defender's mixed strategy (i.e., randomized allocation of security resources). In order to avoid non-compliant participants, we only allowed workers whose HIT approval rates were greater than 95% and who had more than 100 approved HITs to participate in the experiment.

Let G denote a game instance, which is a combination of a payoff structure $\{(R_i^a, P_i^a, R_i^d, P_i^d), t_i \in T\}$, and a defender's strategy x. Given a game instance G, we denote the choice of the j^{th} subject as $\tau_j^G \in T$. We include seven payoff structures in the experiments: four of which are selected based on using a classification method we explain in detail in Section 4.3.1; the other three are taken directly from Pita et al.[Pita et al., 2010]. For each payoff structure we tested five different defender strategies. This results in $7 * 5 = 35$ different game instances. Each of the subjects played all 35 games. In total, 80 subjects participated in the preliminary experiment.

4.3.1 Selecting Payoff Structures

Even for a restricted class of games such as security games, there are an infinite number of possible game instances depending on the specific values of the payoffs for each of the targets. Since we cannot conduct experiments on every possible game instance we need a method to select a set of payoffs structures to use in our experiments. Our main criteria for selecting payoffs structures are (1) to select a diverse set of payoff structures that cover different regions in the space of possible security games and (2) to select payoff structures that will differentiate between the different behavioral models (in other words, the models should make different predictions in different test conditions). In the first round our goal was to select game instance that would distinguish between the three key families of prediction methods (BRPT, RPT, BRQR). In the second round of selection we need to further differentiate within the families. Since there is not yet a well-understood method to select such game instances in the literature, we introduce a procedure for making such selections below.

We first sample randomly 1000 different payoff structures, each with 8 targets. R_i^a and R_i^d are integers drawn from $Z^+[1, 10]$; P_i^a and P_i^d are integers drawn from $Z^-[-10, -1]$. This scale

Table 4.1: A-priori defined features

Feature 1	Feature 2	Feature 3	Feature 4
mean($\lvert\frac{R_i^a}{P_i^a}\rvert$)	std($\lvert\frac{R_i^a}{P_i^a}\rvert$)	mean($\lvert\frac{R_i^d}{P_i^d}\rvert$)	std($\lvert\frac{R_i^d}{P_i^d}\rvert$)
Feature 5	Feature 6	Feature 7	Feature 8
mean($\lvert\frac{R_i^a}{P_i^d}\rvert$)	std($\lvert\frac{R_i^a}{P_i^d}\rvert$)	mean($\lvert\frac{R_i^d}{P_i^a}\rvert$)	std($\lvert\frac{R_i^d}{P_i^a}\rvert$)

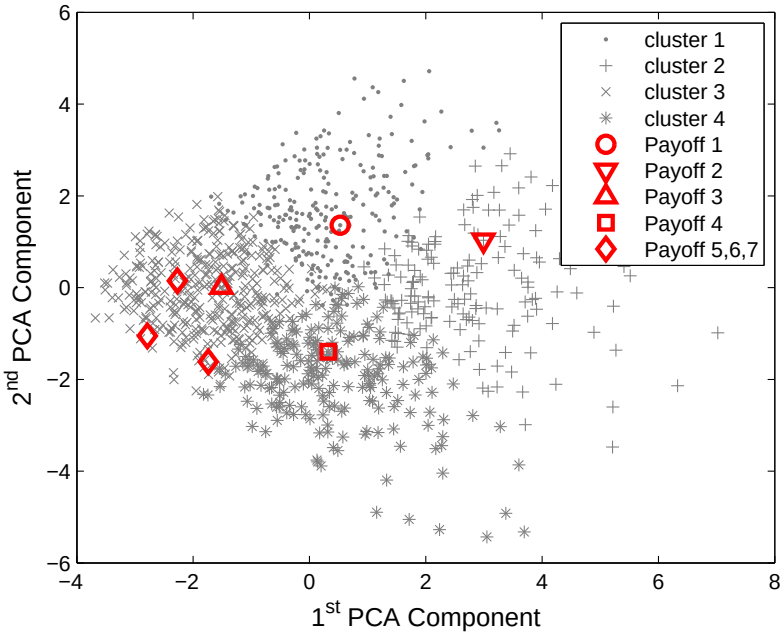


Figure 4.3: Payoff Structure Clusters (color)

is similar to the payoff structures used in [Pita et al., 2010]. We then use k-means clustering to group the 1000 payoff structures into four clusters based on eight features, which are defined in Table 4.1. Intuitively, features 1 and 2 describe how good the game is for the adversary, features 3 and 4 describe how good the game is for the defender, and features 5∼8 reflect the level of conflict between the two players in the sense that they measure the ratio of one player's gain over the other player's loss.

In Fig. 4.3, all 1000 payoff structures are projected onto the first two Principal Component Analysis (PCA) dimensions for visualization. The three payoff structures (5–7) that were first

used in Pita et al.[Pita et al., 2010] are marked in Fig. 4.3. All three of these payoff structures belong to cluster 3, indicating that the game instances used in the previous experiments we all similar in terms of the features we used for classification[2].

To select specific payoff structures from these clusters we first generated five defender strategies based on the following families of algorithms: DOBSS, COBRA, BRPT, RPT and BRQR. Here we select only one algorithm from each family (e.g., only one version of BRQR). At this point we did not have preliminary data to set the parameters of the algorithms, since we are deciding which payoff structures to test on. Instead, we set the parameters as follows: DOBSS has no parameters; for COBRA we use parameters drawn from [Pita et al., 2010]; BRPT and RPT use the empirical parameter settings for Prospect Theory [Kahneman and Tvesky, 1992]; BRQR uses a value of $\lambda = 0.76$ which we set using the data reported in [Pita et al., 2010] (using the method to be described in Section 4.3.3).

We use the following the criteria to select payoff structures that differentiate among the different families of algorithms:

- We define the distance between two mixed strategies, x^k and x^l, using the Kullback-Leibler divergence: $D(x^k, x^l) = D_{KL}(x^k|x^l) + D_{KL}(x^l|x^k)$, where $D_{KL}(x^k|x^l) = \sum_{i=1}^{n} x_i^k \log(x_i^k/x_i^l)$.

- For each payoff structure, $D(x^k, x^l)$ is measured for every pair of strategies. With five strategies, we have 10 such measurements.

[2]In [Pita et al., 2010], there were four payoff structures used, but we only use three of those here. The fourth payoff structure is a zero-sum game, and the deployed Stackelberg security games have not been zero sum [Pita et al., 2008; Tsai et al., 2009]. Furthermore, in zero-sum games, defender's strategies computed from DOBSS, COBRA and MAXIMIN collapse into one – they turn out to be identical.

- We remove payoff structures that have a mean or minimum of these 10 quantities below a given threshold. This results in a subset of about 250 payoff structures in total for all four clusters. We then select one payoff structure closest to the cluster center from each of these subsets.

The four payoff structures (1–4) we selected from different clusters and are marked in Fig. 4.3.

4.3.2 Parameter Estimation for Prospect Theory

An empirical setting of parameter values is suggested in the literature [Kahneman and Tvesky, 1992] based on various experiments conducted with human subjects. We also include this setting of parameter values in our experiments to evaluate the benchmark performance of the prospect theory. At the same time, we provide a method to estimate the parameter values for the PT model using a set of empirical response data collected for the SSG domain. In this section, we describe our method of estimating the parameter values based on using grid search.

The empirical functions we used in the PT model for the adversary have four parameters that must be specified: $\alpha, \beta, \theta, \gamma$, as shown in Equations (4.2) and (4.3). Varying the values for these four parameters will change the responses predicted by the PT-model. We denote the weighting and value function as $\pi_\gamma(\cdot)$ and $V_{\alpha,\beta,\theta}(\cdot)$, for a given a set of parameter values. We then define the fit of a parameter setting to a given data set of subjects' choices as the percentage of subjects who choose the target predicted by the model. The fit can be computed as

$$\texttt{Fit}(\alpha, \beta, \theta, \gamma \mid G) = \frac{1}{N} \sum_{j=1..N} q_{\tau_j^G}(\alpha, \beta, \theta, \gamma \mid G) = \sum_{t_i \in T} \frac{N_i}{N} q_i(\alpha, \beta, \theta, \gamma \mid G)$$

where $q_i(\cdot) \in \{0, 1\}$ indicates whether the PT model predicts target t_i to be chosen by the subjects and is computed using Equation (4.5),N_i is the number of subjects who choose target t_i, and $N = \sum_{t_i \in T} N_i$ is the total number of subjects.

We estimate the parameter setting with the best fit for PT model by maximizing the fit function over all 35 game instances

$$\max_{\alpha,\beta,\theta,\gamma} \quad \sum_{G} \texttt{Fit}(\alpha, \beta, \theta, \gamma \mid G) \tag{4.42}$$

$$s.t. \quad 0 < \alpha, \beta < 1, \theta \geq 1, 0 < \gamma < 1 \tag{4.43}$$

The constraints in (4.43) restrict the feasible range of all the four parameters, as defined in the prospect theory model. The objective function in Equation (4.42) cannot be expressed as a closed-form expression of α, β, θ and γ. Without a closed form it is difficult to apply gradient descent or any other analytical search algorithm to find the optimal solution. Therefore, we use grid search [Sen and Stoffa, 1995; Becsey et al., 1968] to solve the problem as follows:

(1) We first uniformly sample a set of values for each parameter across the feasible ranges, with the following grid intervals: $\Delta_\alpha = 0.05$, $\Delta_\beta = 0.05$, $\Delta_\gamma = 0.05$, and $\Delta_\theta = 0.1$. This gives a set of different values for each of the four parameters. For simplicity, we represents the four sets of sampled values as the following: $\{\alpha_{k_1} = \alpha_l + k_1 \cdot \Delta_\alpha\}$, where α_l is the lower bound of the region; similarly $\{\beta_{k_2} = \beta_l + k_2 \cdot \Delta_\beta\}$; $\{\theta_{k_3} = \theta_l + k_3 \cdot \Delta_\theta\}$; and $\{\gamma_{k_4} = \gamma_l + k_4 \cdot \Delta_\gamma\}$. The feasible region of θ does not have upper bound, so we set it to 5 which is twice as the suggested empirical value [Kahneman and Tvesky, 1992].

(2) In total, we have $20 \cdot 20 \cdot 20 \cdot 40 = 320k$ different combinations of the four parameter values. We then evaluate the objective function on each of the combinations $(\alpha_{k_1}, \beta_{k_2}, \theta_{k_3}, \gamma_{k_4})$ and take the parameter combination with the best aggregate fit as the solution:

$$(\alpha^*, \beta^*, \theta^*, \gamma^*) = \arg \max_{k_1,k_2,k_3,k_4} \sum_G \texttt{Fit}(\alpha_{k_1}, \beta_{k_2}, \theta_{k_3}, \gamma_{k_4} \mid G)$$

The parameter settings estimated using the method described above are:

$$(\alpha^*, \beta^*, \theta^*, \gamma^*) = (1.0, 0.6, 2.2, 0.6)$$

4.3.3 Parameter Estimation for the QR Model

We now explain how we estimate the parameter for the Quantal Response Model (QR Model). The parameter λ in the QR model represents the level of noise in the adversary's response function. We employ Maximum Likelihood Estimation (MLE) to fit λ using data we collected. Given a game instance G and N samples of the subjects' choices $\{\tau_j(G), j = 1..N\}$, the likelihood of λ is

$$L(\lambda \mid G) = \prod_{j=1..N} q_{\tau_j^G}(\lambda \mid G)$$

where, $\tau_j^G \in T$ denotes the target attacked by the j^{th} player and $q_{\tau_j^G}(\lambda \mid G)$ can be computed by Equation (4.7). For example, if player j attacks target t_3 in game G, we would have $q_{\tau_j^G}(\lambda \mid G) = q_3(\lambda \mid G)$. Furthermore, the log-likelihood of λ is

$$\log L(\lambda \mid G) = \sum_{j=1}^{N} \log q_{\tau_j(G)}(\lambda \mid G) = \sum_{t_i \in T} N_i \log q_i(\lambda)$$

Combining with Equation (4.6),

$$\log L(\lambda \mid G) = \lambda \sum_{t_i \in T} N_i U_i^a(x_i) - N \log(\sum_{t_i \in T} e^{\lambda U_i^a(x)})$$

We learn the optimal parameter setting for λ by maximizing the total log-likelihood over all 35 game instances:

$$\max_{\lambda} \quad \sum_{G} \log L(\lambda \mid G) \tag{4.44}$$

$$s.t. \quad \lambda \geq 0 \tag{4.45}$$

The objective function in Equation (4.44) is concave, since for each G, a $\log L(\lambda \mid x)$ is a concave function. This can be demonstrated by showing that the second order derivative of $\log L(\lambda \mid G)$ is non-positive $\forall G$:

$$\frac{d^2 \log L}{d\lambda^2} = \frac{\sum_{i<j} -(U_i^a(x_i) - U_j^a(x_j))^2 e^{\lambda(U_i^a(x_i)+U_j^a(x_j))}}{(\sum_i e^{\lambda U_i^a(x_i)})^2} \leq 0$$

Therefore, $\log L(\lambda \mid x)$ only has one local maximum. We use gradient descent solve the above optimization problem. The MLE of λ is

$$\lambda^* = 0.55$$

.

4.3.4 Parameter Estimation for the QRRU Model

For the QRRU Model, we need to estimate two parameters: λ_u and λ_s as defined in Equation (4.8). We again apply Maximum Likelihood Estimation, similar to the method for the QR model. Given a game instance G, and the responses of N subjects $\{\tau_j(G), j = 1..N\}$, the log-likelihood of a parameter setting (λ_u, λ_s) is

$$\log L(\lambda_u, \lambda_s \mid G) = \sum_{j=1}^{N} \log q_{\tau_j(G)}(\lambda_u, \lambda_s \mid G) = \sum_{t_i \in T} N_i \log q_i(\lambda_u, \lambda_s)$$

Combining with Equation (4.8),

$$\log L(\lambda_u, \lambda_s \mid G) = \lambda_u \sum_{t_i \in T} N_i U_i^a(x_i) + \lambda_s \sum_{t_i \in T} N_i S_i(x) - N \log(\sum_{t_i \in T} e^{\lambda_u U_i^a(x_i) + \lambda_s S_i(x)})$$

We learn the optimal parameter settings for the QRRU Model by maximizing the total log-likelihood over all 35 game instances:

$$\max_{\lambda_u, \lambda_s} \quad \sum_G \log L(\lambda_u, \lambda_s \mid G) \tag{4.46}$$

$$s.t. \quad \lambda_u \geq 0, \lambda_s \geq 0 \tag{4.47}$$

The objective function in Equation (4.46) is a concave function, since $\forall G$ the Hessian matrix of $\log L(\lambda_u, \lambda_s \mid G)$ is negative semi-definite. We include the details of proof in the appendix and only show here that $\forall \langle \lambda_u, \lambda_s \rangle$

$$\langle \lambda_u, \lambda_s \rangle \cdot H(\lambda_u, \lambda_s \mid G) \cdot \langle \lambda_u, \lambda_s \rangle^T \leq 0$$

where $H(\lambda_u, \lambda_s \mid G)$ is the Hessian matrix of $\log L(\lambda_u, \lambda_s \mid G)$ computed as the following

$$H(\lambda_u, \lambda_s \mid G) = -N \begin{pmatrix} \frac{\sum_{i<j}(U_i^a - U_j^a)^2 e^{A_i + A_j}}{(\sum_{t_i \in T} e^{A_i})^2} & \frac{\sum_{i<j}(U_i^a - U_j^a)(S_i - S_j) e^{A_i + A_j}}{(\sum_{t_i \in T} e^{A_i})^2} \\ \frac{\sum_{i<j}(U_i^a - U_j^a)(S_i - S_j) e^{A_i + A_j}}{(\sum_{t_i \in T} e^{A_i})^2} & \frac{\sum_{i<j}(S_i - S_j)^2 e^{A_i + A_j}}{(\sum_{t_i \in T} e^{A_i})^2} \end{pmatrix}$$

where, $A_i = \lambda_u U_i^a(x_i) + \lambda_s S_i(x)$. Therefore, we can use gradient descent to solve the optimization problem in Equation (4.46) and (4.47). The MLE parameters based on our data set are:

$$(\lambda_u^*, \lambda_s^*) = (0.6, 0.77)$$

.

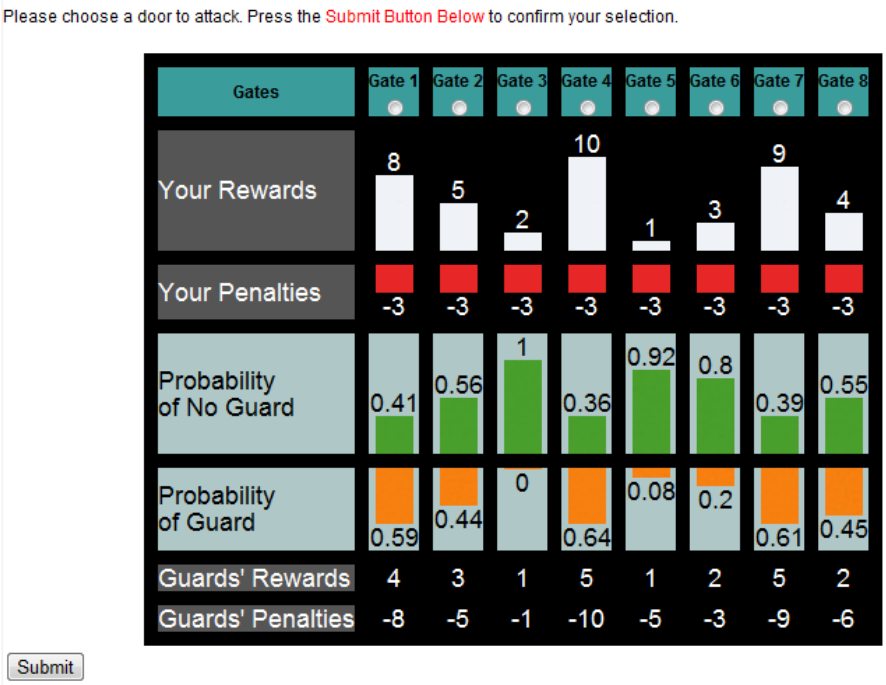


Figure 4.4: Game interface for our simulated online SSG

4.4 Experimental Results and Discussion

We evaluated the performances of defender strategies as well as the the accuracy of different adversary models with human subjects using an online game "The Guard and The Treasure" that will be introduced soon. We conducted two set of evaluations: the first set includes the same 7 payoff structures used in the experiments in the previous section; the second set focuses on comparison between the QR model and the QRRU model.

4.4.1 A Simulated Online SSG

We develop a game, called "The Guards and The Treasure", to simulate the security model at the LAX airport, which has eight terminals that can be targeted in an attack [Pita et al., 2008]. Fig. 6.1 shows the interface of the game. Players are introduced to the game through a series of explanatory screens describing how the game is played. In each game instance a subject is

asked to choose one of the eight gates to open (attack). They are told that guards are protecting three of the eight gates, but not which ones. The defender's mixed strategy, represented as the marginal probability of covering each target, $< x_i >$, is given to the subjects. At the same time, the subjects are also told the reward on successfully attacking each target as well as the penalty of getting caught at each target. The three gates protected by the guards are drawn randomly from the probability shown on the game interface. If subjects select a gate protected by the guards, they receive a penalty; otherwise, they receive a reward. Subjects are rewarded based on the reward/penalty shown for each gate. For example, in the game shown in Figure 6.1, the probability that gate 1 (target 1) will be protected by a guard is 0.59. Assuming the subjects choose gate 1, he/she gets reward of 8 if gate 1 is not protected by the guard; or get a penalty of -3 if gate 1 is protected by a guard.

4.4.2 Experimental Settings

The design of the simulated game was already provided in Section 4.4.1. We now present a detailed description of the experimental settings. In total, we included 70 game instances (comprising 7 payoff structures and 10 strategies for each payoff structure) in the first set and 12 game instances (comprising 4 new payoff structures and 3 strategies for each payoff structure) in the second set. To avoid confusion between these two sets of payoff structures, we will number the first seven payoff structures as 1.1-1.7, and the next four as 2.1-2.4.

Each game instance is played by at least 80 different participants (the actual number of subjects for each game instance ranges between 80 to 91). Each subject is asked to play 40 out of the 70 games. For the purpose of a within-subject comparison, we want a subject to play the 10 different strategies for the same payoff structure. Therefore, the 40 games is composed of 4

payoff structures and 10 defender strategies for each. Furthermore, in order to mitigate the ordering effect on subject responses, we randomize the order of the game instances played by each subject. We generated 40 different orderings of the games using latin square design. The order played by each subject was drawn uniformly randomly from the 40 possible orderings. To further mitigate ordering effect, no feedback on success or failure is given to the subjects until the end of the experiment. As motivation to the subjects, they earn or lose money based on whether or not they succeed in attacking a gate; if the subject opens a gate not protected by the guards, they win; otherwise, they lose.

The participants were recruited on Amazon Mechanical Turk. Note that these participants differ from those who played the game to provide data for estimating the parameter, as discussed in the previous section. In order to avoid non-compliant participants, we only allowed workers whose HIT approval rates were greater than 95% and who had more than 100 approved HITs to participate in the experiment. They were first given a detailed instruction of the game explaining to them how the game is played. Then two practical rounds of games were provided to help them get familiar with the game. After all the learning and practising, they were given enough time to finish all the games.

Each participant first received 50 cents for participating in the game. Then they gain bonus based on the outcomes of the games they played, with each point worth 1 cent. On average, the subjects who participated in the first set of experiment (i.e. payoff 1.1-1.7) received $\$1.45$ as bonus based on their total scores across 40 game instances they played; the subjects who participated in the second set of experiment (i.e. payoff 2.1-2.4) received $\$0.44$ as bonus based on their total scores across 12 game instances they played. Participants were given 5 hours in total to finish the experiment which was shown to be sufficiently long given that the average time

Payoff	1.1	1.2	1.3	1.4	1.5	1.6	1.7
COBRA-α	0.15	0.15	0.15	0.15	0.37	0	0.25
COBRA-ϵ	2.5	2.9	2.0	2.75	2.5	2.5	2.5
BRPT-E	$(\alpha, \beta, \theta, \gamma) = (0.88, 0.88, 2.25, 0.64)$						
RPT-E	$(\alpha, \beta, \theta, \gamma) = (0.88, 0.88, 2.25, 0.64)$, $\epsilon = 2.5$						
BRPT-L	$(\alpha, \beta, \theta, \gamma) = (1, 0.6, 2.2, 0.6)$						
RPT-L	$(\alpha, \beta, \theta, \gamma) = (1, 0.6, 2.2, 0.6)$, $\epsilon = 2.5$						
BRQR-76	$\lambda = 0.76$						
BRQR-55	$\lambda = 0.55$						
BRQRRU	$(\lambda_u, \lambda_s) = (0.6, 0.77)$						

Table 4.2: Parameter settings for different algorithms

they spent was 28 minutes for the first set of 40 games and 8 minutes for the second set of 12 games.

In the following part of this section, we first describe the parameter settings for the different leader strategies. We then provide our experimental results, and follow that up with analysis. We compare both the quality of different defender strategies against the human participants and the accuracy of different adversary models in the sense that how well the human participants follow the assumption of these models.

4.4.3 Algorithm Parameters

For the seven payoff structures (1.1-1.7) introduced in Section 4.3, we tested ten different mixed strategies generated from seven different algorithms: MAXIMIN, DOBSS [Paruchuri et al., 2008], COBRA [Pita et al., 2010], BRPT, RPT, BRQR, BRQRRU. We include MAXIMIN as a benchmark algorithm. MAXIMIN assumes that adversary always selects the target that is worst to the defender. Table 4.2 lists the parameter settings of these ten strategies for each of the seven payoff structures.

- DOBSS and MAXIMIN have no parameters.

- For COBRA, we set the parameters following the methodology presented in [Pita et al., 2010] as closely as possible for payoff structures 1.1$\sim$1.4,. In particular, the values we set for α meet the entropy heuristic discussed in that work. For payoff structures 1.5$\sim$1.7 that are identical to payoff structures first used by Pita et al., we use the same parameter settings as in their work.

- For both BRPT-E and RPT-E, the parameters for Prospect Theory are empirical values suggested by literatures [Kahneman and Tvesky, 1992]. For RPT-E, we empirically set ϵ to 25% of the maximum potential reward for the adversary, which is 10 in our experimental settings.

- We tried another set of parameters for Prospect Theory, which are learned from our first set of experiment as described in Section 4.3.2. We denote these two algorithms as BRPT-L and RPT-L.

- For BRQR, we tried two different values for the parameter λ, $\lambda = 0.76$ is the values learned from the data reported by Pita et al.[Pita et al., 2010]; $\lambda = 0.55$ is the value learned from data collected in our first set of experiments with participants from Amazon Mechanical Turk. We will refer to the strategies resulting from these two parameter settings of the BRQR algorithm as BRQR-76 and BRQR-55 respectively.

- For BRQRRU, the parameters are learned from the data collected our first set of experiments.

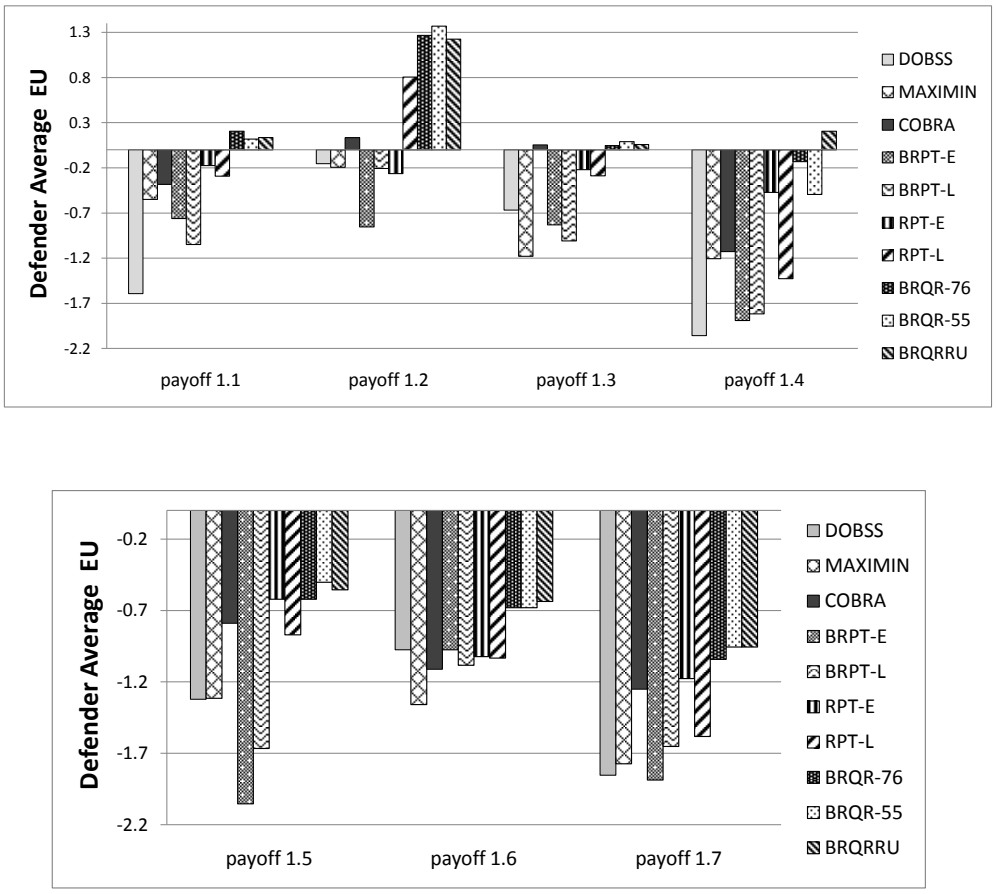


Figure 4.5: Defender average expected utility achieved by different strategies

4.4.4 Quality Comparison

We evaluated the performance of different defender strategies using the defender's expected utility and the statistical significance of our results using the bootstrap-t method [Wilcox, 2003].

4.4.4.1 Average Performance

We first evaluated the average defender expected utility, $U^d_{avg}(x)$, of different defender strategies based on the subjects' choices:

$$U^d_{avg}(x) = \frac{1}{N}\sum_{j=1}^{N} U^d_{\tau_j}(x) = \frac{1}{N}\sum_{t_i \in T} N_i U^d_i(x_i)$$

where τ_j is the target selected by the j^{th} subject, N_i is the number of subjects that chose target t_i and N is the total number of subjects. Fig. 4.5 displays $U^d_{avg}(x)$ for the different strategies in each payoff structure. We also displayed the normalized defender average expected utility of different strategies within each payoff structure in Figure 4.6. After normalization, $U^d_{avg}(x)$ for each defender strategy varies between 0 and 1, with the highest $U^d_{avg}(x)$ in each payoff structure scaled to 1 and the lowest $U^d_{avg}(x)$ scaled to 0.

Overall, BRQR-76, BRQR-55 and BRQRRU performed better than other algorithms. We compare the performance of three algorithms with each of the other seven algorithms and report the level of statistical significance in Table 4.3, 4.4 and 4.5. We summarize the results below:

- MAXIMIN is outperformed by all three algorithms with statistical significance in all seven payoff structures. DOBSS is also outperformed by all three algorithms with statistical significance except for payoff structure 1.6.

- In five of the seven payoff structures, COBRA is outperformed by all three algorithms with statistical significance. In payoff structure 1.3, the performance of COBRA is very close to the three algorithms, but there is no statistical significance either way. In payoff structure 1.5, COBRA is outperformed by all three algorithms but no statistical significance is achieved.

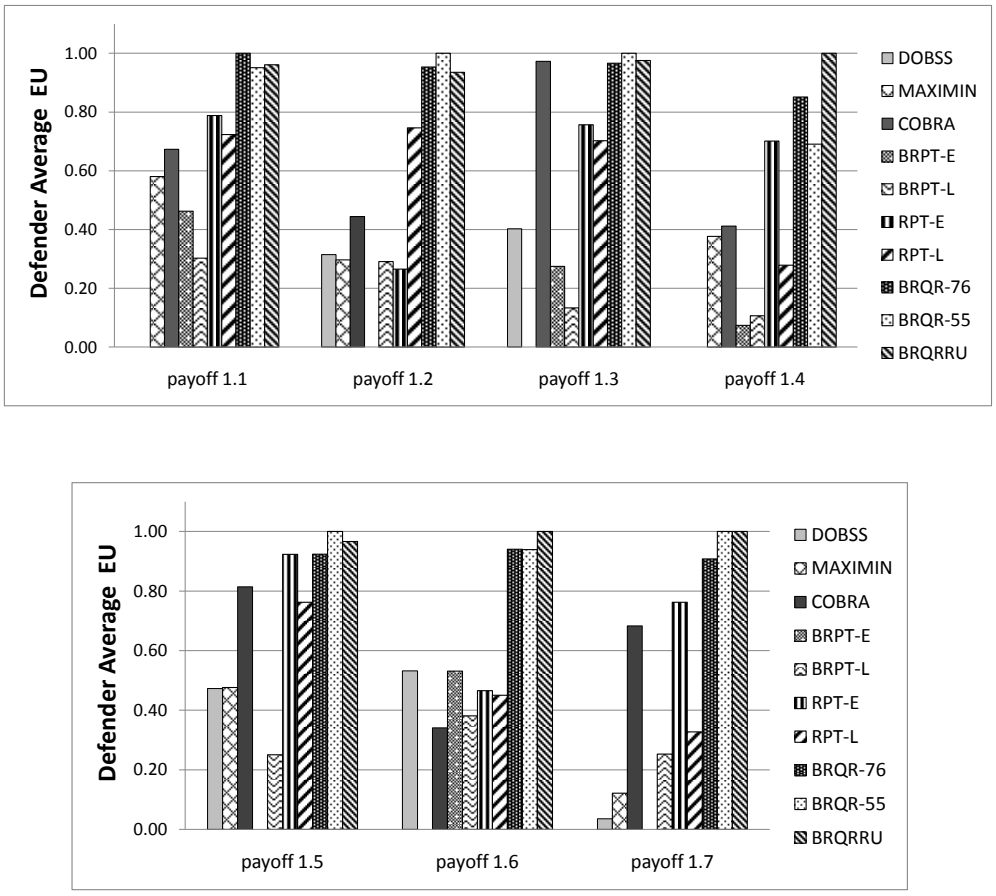


Figure 4.6: Defender average expected utility (normalized between 0 and 1) achieved by different strategies

v.s.	DOBSS	MAXIMIN	COBRA	BRPT-E	RPT-E	BRPT-L	RPT-L
payoff 1.1	***	***	***	***	**	***	***
payoff 1.2	***	***	***	***	***	***	0.15
payoff 1.3	***	***	0.96	***	0.21	***	**
payoff 1.4	***	***	*	***	0.25	***	***
payoff 1.5	***	***	0.26	***	0.99	***	***
payoff 1.6	0.20	***	***	*	***	0.13	***
payoff 1.7	***	***	**	***	**	***	***

Table 4.3: level of statistical significance of comparing BRQR-76 to other algorithms: ***($p \leq 0.01$), **($p \leq 0.05$), *($p \leq 0.1$)

- The three algorithms outperform BRPT-E with statistical significance in all seven payoff structures. Furthermore, BRPT-L is outperformed by the three algorithms in all seven payoff structures with statistical significance in six cases except for in payoff structure 1.6.

- In four of the seven payoff structures, RPT-E is outperformed by the three algorithms with statistical significance. In payoff 1.3, RPT-E is outperformed by all three algorithms but the result is not statistical significant. In payoff structure 1.4, RPT-E achieves very similar performance to BRQR-55 and is outperformed by BRQR-76 and BRQRRU. In payoff 1.5, RPT-E achieves very similar performance as BRQR-76 and is outperformed by BRQR-55 and BRQRRU. Furthermore, RPT-L is outperformed by all three algorithms with statistical significance in almost all seven payoff structures, except for in payoff structure 1.2 where the result of comparing BRQR-76 and BRQRRU with RPT-L doesn't have statistical significance.

Overall, any of the three quantal response (BRQR-76,BRQR-55 and BRQRRU) strategies would be preferred over the other strategies. However, the performance of the three strategies are close to each other in this set of experiments. In order to further differentiate the three strategies as well as prove the effectiveness of QRRU model, we conducted a separate set of experiments.

v.s.	DOBSS	MAXIMIN	COBRA	BRPT-E	RPT-E	BRPT-L	RPT-L
payoff 1.1	***	***	**	***	*	***	***
payoff 1.2	***	***	**	***	***	***	*
payoff 1.3	***	***	0.86	***	0.16	***	**
payoff 1.4	***	***	**	***	0.95	***	**
payoff 1.5	***	***	0.37	***	0.12	***	**
payoff 1.6	0.16	***	***	**	***	0.11	***
payoff 1.7	***	***	***	***	***	***	***

Table 4.4: level of statistical significance of comparing BRQR-55 to other algorithms: ***($p \leq 0.01$), **($p \leq 0.05$), *($p \leq 0.1$)

v.s.	DOBSS	MAXIMIN	COBRA	BRPT-E	RPT-E	BRPT-L	RPT-L
payoff 1.1	***	***	**	***	*	***	***
payoff 1.2	***	***	**	***	***	***	0.27
payoff 1.3	***	***	0.99	***	0.27	***	**
payoff 1.4	***	***	**	***	0.18	***	***
payoff 1.5	***	***	0.40	***	0.33	***	*
payoff 1.6	0.15	***	***	**	***	0.11	***
payoff 1.7	***	***	***	***	***	***	***

Table 4.5: level of statistical significance of comparing BRQRRU to other algorithms: ***($p \leq 0.01$), **($p \leq 0.05$), *($p \leq 0.1$)

We first select four new payoff structures from the 1000 random samples using the following rules::

- We first measure the distance between the BRQRRU strategy and each of the other two BRQR strategies using Kullback-Leibler (KL) divergence: $D(x^k, x^l) = D_{KL}(x^k|x^l) + D_{KL}(x^l|x^k)$, where $D_{KL}(x^k|x^l) = \sum_{i=1}^{n} x_i^k \log(x_i^k/x_i^l)$.

- For each payoff structure, we measure this KL distance for the pair (BRQRRU,BRQR-76) and the pair (BRQRRU,BRQR-55). So we have two such measurements for each payoff structure.

- We sort the payoff structures in a descending order of the mean of these two distance.

- In the top 10 payoff structures, we select two payoff structures where the targets assigned with minimum coverage probability by BRQR-76 or BRQR-55 have large penalty for the defender; and two payoff structures where the penalty for the defender on such target is small.

The details of these four payoff structures and the defender strategies are included in the appendix. We conducted a new set of experiments with human subjects using these four payoff structures and the three QR model based strategies for each payoff structure. In total, we have 4*3 = 12 game instances included in these experiments. Each subject is asked to play against all these 12 game instances. 80 subjects are involved in these experiments.

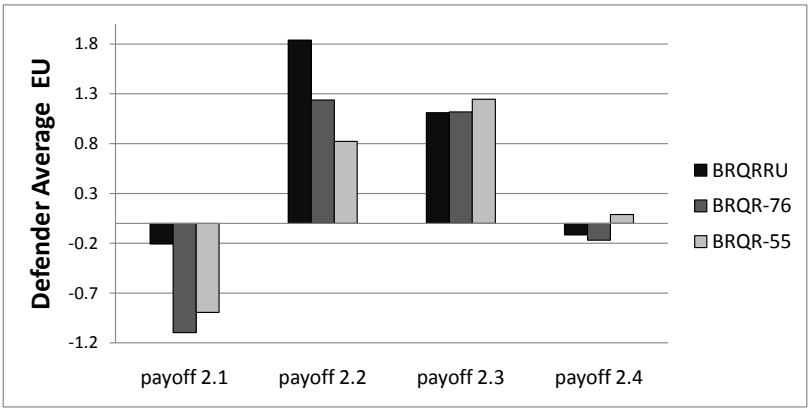


Figure 4.7: Defender average expected utility achieved by QR model based strategies

Figure 4.7 displays the defender average expected utility achieved by the three strategies. We report the statistical significance results in Table 4.6. In payoff structures 2.1 and 2.2, BRQRRU outperforms both BRQR-76 and BRQR-55 with statistical significance. In payoff structures 2.3 and 2.4, the three strategies have very close performance. No statistical significance is found in the results, as reported in Table 4.6.

payoff 2.1	BRQRRU v.s. BRQR-76	***
	BRQRRU v.s. BRQR-55	**
payoff 2.2	BRQRRU v.s. BRQR-76	**
	BRQRRU v.s. BRQR-55	**
payoff 2.3	BRQR-76 v.s. BRQRRU	0.87
	BRQR-55 v.s. BRQRRU	0.40
payoff 2.4	BRQRRU v.s. BRQR-76	0.97
	BRQR-55 v.s. BRQRRU	0.35

Table 4.6: statistical significance (**: $p \leq 0.05$; ***: $p \leq 0.01$)

As noted earlier, a very important feature of payoff structures 2.1 and 2.2, compared to payoff structures 2.3 and 2.4, is that the target covered with minimum resource by BRQR-76 and BRQR-55 (target 3 in payoff structure 2.1 and target 3 in payoff structure 2.2) has a large penalty (≤ -6) for the defender. In the experiments with payoff structures 2.1 and 2.2, more than 10% of subjects selected these targets (target 3 in payoff structure 2.1 and 2.2) while playing against BRQR-76 or BRQR-55, while no subjects chose this target while playing against BRQRRU— BRQRRU covers these targets with more resources. This is the main reason why BRQRRU significantly outperforms BRQR in payoff 2.1 and payoff 2.2. In payoff 2.3 and 2.4, similar observation is obtained in subjects' choice: the targets covered with minimum resources by BRQR-76 and BRQR-55 are selected more frequently compared to the case when BRQRRU is played. However, these targets (i.e. target 1 in payoff 2.3 and target 2 in payoff 2.4) have very small penalty for the defender (-1). Therefore we do not see significant differences in performance among the different BRQR strategies.

Based on the result in both sets of experiments, we conclude that the stochastic model based strategies are superior to their competitors, and BRQRRU is the preferred strategy within the stochastic model based strategies.. In particular, BRQRRU achieves significantly better performance than BRQR when the target covered with minimum resource by BRQR has potentially a

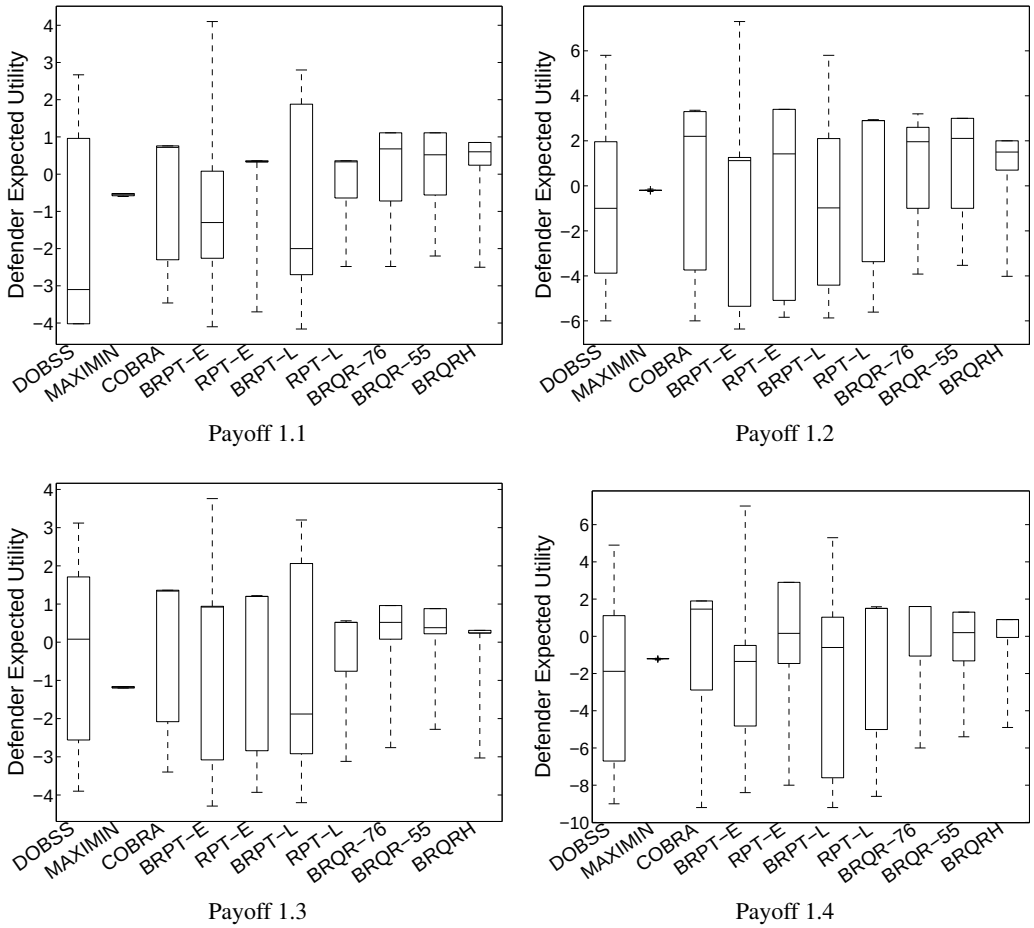
large penalty for the defender; and has a performance similar to the other stochastic model based strategies otherwise.

4.4.4.2 Performance Distribution

We now analyze the distribution of the performance of each defender strategy while playing against different adversaries (subjects). Given a game instance G, the defender expected utility achieved by playing strategy x against a subject j is denoted as $U^d_{\tau^G_j}(x)$. Figures 4.8 and 4.9 display the distribution of $U^d_{\tau^G_j}(x)$ for different defender strategies against individual subjects in each payoff structure. The y-axis shows the range of the defender's expected utility against all different subjects. Each box with the extended dash line in the figure shows the distribution of this defender expected utility for each of the ten defender strategies: the dashed line specifies the range of $U^d_{\tau^G_j}(x)$ with the bottom band showing the minimum value and the top band showing the maximum value; the box specified the 25^{th} to 75^{th} percentiles of $U^d_{\tau^G_j}(x)$ with the bottom showing the 25^{th} percentile value and the top showing the 75^{th} value; the band inside the box specifies the median (50^{th} percentile) of $U^d_{\tau^G_j}(x)$. We compare the distributions of different defender strategies from two perspectives:

Range: As presented in Figure 4.8 and Figure 4.9, in general, the defender expected utility has the smallest range when MAXIMIN strategy is played (except that in payoff structure 1.7, the range of the defender expected utility when RPT-L is played is slightly smaller than that when MAXIMIN is played). COBRA, RPT, BRQR and BRQRRU lead to larger range of defender expected utility than MAXIMIN. Defender expected utility has the largest range when DOBSS or BRPT is played.

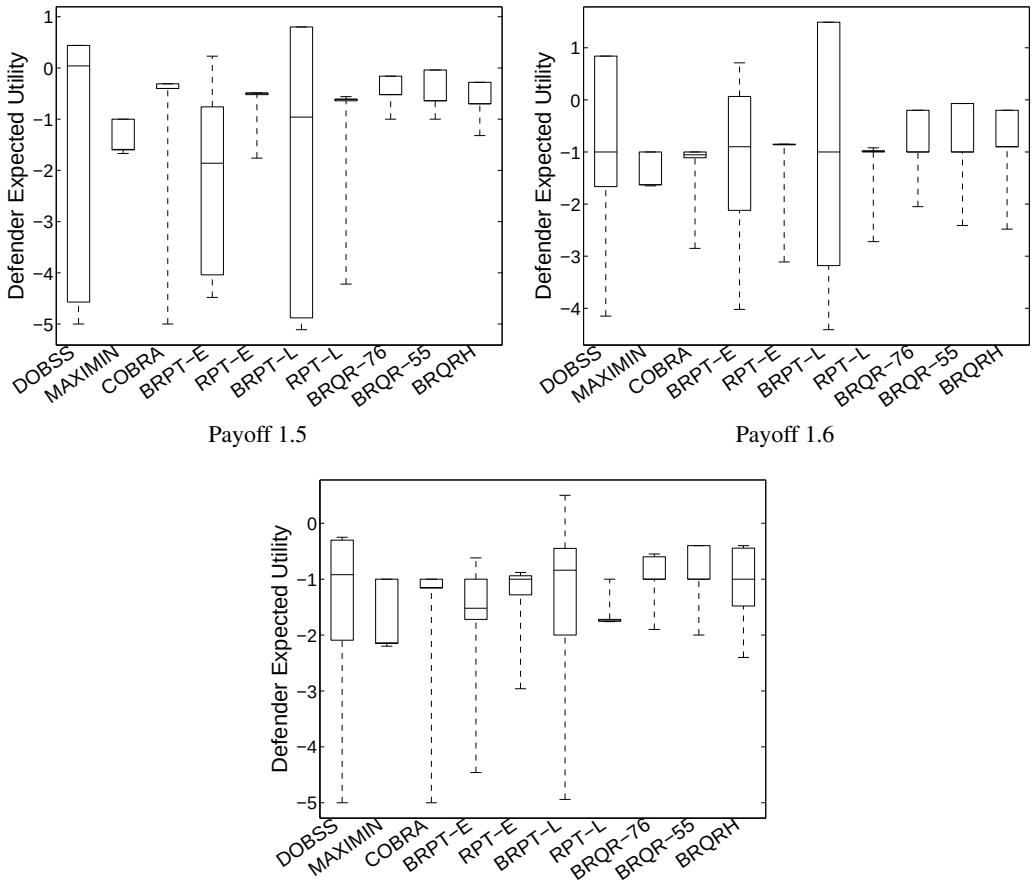
Worst Case: The lower band of the dashed line indicates the worst-case defender expected utility when different strategies are played. MAXIMIN has the highest worst-case defender expected utility in general (except that in payoff 1.5, the worst-case defender expected utility by playing BRQR-76 is better than that by playing MAXIMIN). DOBSS and BRPT lead to lowest worst-case defender expected utility. The worst-case defender expected utility from playing COBRA, RPT, BRQR and BRQRRU are in between the two extreme cases. Furthermore, BRQR and BRQRRU lead to higher worst-case defender expected utility than COBRA and RPT.



Payoff 1.1 Payoff 1.2

Payoff 1.3 Payoff 1.4

Figure 4.8: Distribution of defender's expected utility against each individual subject



Payoff 1.5

Payoff 1.6

Payoff 1.7

Figure 4.9: Distribution of defender's expected utility against each individual subject

In general, by playing MAXIMIN, the defender expected utility against each individual adversary achieves the smallest variance, hence it is most robust to the uncertainty in adversary's choice. However, it does so by assuming that the adversary could select any target hence making the expected utility on each target equal. MAXIMIN does not exploit the different preferences adversary may have among different targets. BRPT and DOBSS assume the subjects select the target that maximizes their expected utility and do not consider the possibility of deviations from the optimal choice by the adversary. This leads to arbitrarily lower defender expected utility when the adversary deviates from the predicted choice.

COBRA, RPT, BRQR and BRQRRU all try to be robust against such deviations. BRQR and BRQRRU consider some (possibly very small) probability of adversary attacking any target using a soft-max function. In contrast, COBRA and RPT separate the targets into two groups, the ϵ-optimal set and the non-ϵ-optimal set, using a hard threshold. They then try to maximize the worst case for the defender assuming the response will be in the ϵ-optimal set, but assign less resources to the non-ϵ-optimal targets. When the non-ϵ-optimal targets have high defender penalties, COBRA and RPT become vulnerable to adversary's deviation. For example, target 6 in payoff structure 1.2 has a small reward ($= 1$) and a large penalty ($= -10$) for the attacker. Both COBRA and RPT consider this target to be in the non-ϵ-optimal set and assign very small probability to cover this target (≤ 0.05). However, approximately 10% of the subjects have chosen this target. Since this target has a high defender penalty (-6), COBRA and RPT lose reward on this target. Similar examples include target 5 in payoff structure 1.4 and target 8 in payoff structure 1.1.

4.4.5 Model Prediction Accuracy

In this section, we evaluate how well each model predicts the actual responses of human participants using three different metrics [Feltovich, 2000]: mean square deviation (*MSD*), a proportion of inaccuracy (*POI*), and Euclidean distance (*ED*).

We first extend the definition of *MSD* from that in [Feltovich, 2000] which is designed for a 2-action game, in order to suit our domain where the player has 8 actions to take. Given the choices of the N subjects, the *MSD* of a model is computed as

$$MSD = \{\frac{1}{N}\sum_{n=1}^{N}(p_{\tau(n)} - 1)^2\}^{1/2} \tag{4.48}$$

where, $\tau(n)$ represents the index of the target chosen by subject n, p_i is the predicted probability by a model that target i will be chosen.

The *POI* score is meant to put models with deterministic prediction on the same footing as those with stochastic prediction. It treats the target with the highest predicted probability as the predicted target, and computes the proportion of the subjects who didn't choose the predicted target. The *POI* score is computed as

$$POI = \frac{1}{N}\sum_{n=1}^{N}(1 - \tilde{p}_{\tau(n)}) \tag{4.49}$$

where, $\tau(n)$ is the index of the target chosen by subject n. $\tilde{p}_{\tau(n)} = 1$ if $\tau(n)$ is the predicted target; and $\tilde{p}_{\tau(n)} = 0$ otherwise. Note that for models with deterministic prediction, the *POI* score is exactly equal to the square of *MSD* value.

The Euclidean distance measures the difference between the actual distribution of the subjects' choices and the prediction of the model. It is computed as

$$ED = \sqrt{\sum_{i \in T}(p_i - p_i^{act})^2} \tag{4.50}$$

where p_i is the probability predicted by the model that target i will be chosen, and p_i^{act} is the actually percentage of subjects who have chosen target i.

Table 4.7 presents the ability of different models to predict the attacker decision measured with the three different criteria[3]. The measurements for both the out-of-sample data (70 rounds of games) and in-sample data (35 rounds of games) are displayed in the table. Better predictive power is indicated by lower *MSD* value and *POI* score and lower *ED* value. The top four models all have deterministic prediction and the three quantal response related models have stochastic prediction. The last three models (COBRA, RPT-E and RPT-L) don't have a strict definition of the

[3] MAXIMIN doesn't have a prediction of adversary behavior, so we exclude it from the analysis.

Table 4.7: Ability of behavioral models to predict attacker decision

	Out of sample			In sample		
Model	*MSD*	*POI*	*ED*	*MSD*	*POI*	*ED*
DOBSS	0.81	0.67	0.76	0.85	0.73	0.80
PT-E	0.84	0.71	0.81	0.87	0.75	0.84
PT-L	0.84	0.71	0.81	0.86	0.74	0.83
QR-76	0.79	0.67	0.23	0.83	0.73	0.22
QR-55	0.81	0.67	0.22	0.84	0.73	0.21
QRRU	0.80	0.65	0.21	0.83	0.70	0.18
COBRA	0.91	0.83(0.35)	0.94	0.91	0.83(0.42)	0.93
RPT-E	0.93	0.87(0.52)	0.99	0.94	0.88(0.56)	0.99
RPT-L	0.93	0.86(0.49)	0.98	0.93	0.86(0.54)	0.96

prediction of the attacker's behavior. They are modifications of the base models for robustness. For example, COBRA modifies DOBSS by assuming that attacker will deviate from choosing the target with the highest expected utility to any other targets whose expected utilities are within ϵ of the highest value. However, within this subset of possibly chosen targets, the model doesn't explicitly predict the behavior of the attacker but rather plays a maximin strategy (i.e. maximizing the lowest expected utility). RPT-E and RPT-L modify PT-E and PT-L in similar ways. Given the above property of these three models, we compute the *POI* score in two different ways by using two different definitions of the model prediction.

- The first definition predicts a single target with the lowest expected utility for the defender within the subset of possible deviations. Therefore the *POI* score counts the proportion of subjects who have chosen any other targets.

- The second definition predicts all the targets within the subsect of the possible deviations. Therefore, the *POI* score only counts for the targets outside this subset.

The *POI* score computed with the first definition should be equal to or higher than the value computed with the second definition. Note that the second definition doesn't satisfy the property

of prediction since the sum of the predictions on all targets might be larger than 1. We use this definition to mainly show the importance of accounting for deviation of attackers' decision. The *POI* values computed with the second definition are shown in parentheses in Table 4.7. The observations from the table are summarized blow,

1. For the out-of-sample data, less than 30% of the subjects have selected the target predicted by PT-E or PT-L; in the other words, more than 70% of the subjects have deviated from the prediction. For DOBSS, on average 67% of the subjects deviated from the predicted response. Similar patterns can be observed for the in-sample data.

2. Both RPT and COBRA take into consideration the deviation of the subjects' responses from their optimal action. The percentage of subjects deviate from the model prediction decreased significantly: for the out-of-sample data, the *POI* score of COBRA is 0.35 compared to 0.67 of DOBSS; the *POI* score of RPT-E decreased by 0.19 compared to PT-E; the *POI* score of RPT-L decreased by 0.22 compared to PT-L. Similar patterns are observed for the in-sample data.

3. The *POI* score of QR-76 and QR-55 is the same as DOBSS. This is expected since the target predicted by the QR model to be chosen with the highest probability is the target with the highest expected utility for the attacker, which is the also prediction of DOBSS. In other words, QR-76 and QR-55 have the same predicted target as DOBSS. At the same time, QRRU has the lowest *POI* score among all the models in both the out-of-sample data and in-sample data. The *MSD* scores of the three QR-related models are better (lower) than other models (except that in the out-of-sample data QR-55 has the same score as DOBSS).

4. The advantage of the the three QR related models is most significant under the *ED* score, which represents the error of the model in predicting the distribution of subjects' choices. As shown in Table 4.7, the three QR-related models have significantly lower *ED* scores than the

other models. This is essentially the reason why the three models achieved significantly better defender expected utility than the other models.

Chapter 5: Quantal Response Model with Subjective Utility

In this chapter, I compare the quantal response model to an alternative approach for addressing human bounded rationalitym: a robust optimization approach, which intentionally avoids modeling human decision making. The leading contender here is an algorithm called MATCH ([Pita et al., 2012]). Instead of modeling the particular probabilities of the adversarys deviations from the optimal choice, MATCH only guarantees a bound for the loss to the defender if the adversary deviates from selecting optimally (maximum-expected value choice). It has been shown (in [Pita et al., 2012]) that MATCH significantly outperforms BRQR (Section 4.2) even when significant amounts of human subject data were used to tune the key parameter of the quantal response model. It hence become unclear whether there is still any value in using human behavior models in solving SSG.

In this chapter, using a large number of human subject experiments, I illustrate the importance of integrating human behavior models (and in particular the QR model) within algorithms to solve SSGs. Section 5.1 introduces an extended version of the quantal response model by integrating it with an novel subjective utility function. Section 5.2 provides an improved version of the MATCH algorithm by integrating it with the same subjective utility function learned from the data. Then in section 5.3, I conduct experiments comparing the extended quantal resonse model with

the MATCH algorithm under different settings with human subjects, including both the Amazon Mechanical Turk workers and a group of security intelligence experts.

5.1 The SUQR Model

The key idea in subjective expected utility (SEU) as proposed in behavioral decision-making [Savage, 1972; Fischhoff et al., 1981] is that individuals have their own evaluations of each alternative during decision-making[1]. Recall that in an SSG, the information presented to the human subject for each choice includes: the marginal coverage on target t (x_t); the subject's reward and penalty (R_t^a,P_t^a); the defender's reward and penalty (R_t^d, P_t^d). Inspired by the idea of SEU, we propose a subjective utility function of the adversary for SSG as the following:

$$\hat{U}_t^a = w_1 x_t + w_2 R_t^a + w_3 P_t^a \tag{5.1}$$

The novelty of our subjective utility function is the linear combination of the values (rewards/penalty) and *probabilities*. (Note that we are modeling the decision-making of the general population not of each individual as we do not have sufficient data for each specific subject). While unconventional at first glance, as shown later, this model actually leads to higher prediction accuracy than the classic expected value function. A possible explanation for that is that humans might be driven by simple heuristics in their decision making. Other alternatives to this subjective utility function are feasible, e.g., including all the information presented to the subjects ($\hat{U}_t^a = w_1 x_t + w_2 R_t^a + w_3 P_t^a + w_4 R_t^d + w_5 P_t^d$), which we discuss later.

[1] Similar approach with subjective utility function has been shown to predict human behavior well in previous work [Azaria et al., 2012]

We modify the QR model by replacing the classic expected value function with the SU function, leading to the SUQR model. In the SUQR model, the probability that the adversary chooses target t, q_t, is given by:

$$q_t = \frac{e^{\lambda \hat{U}_t^a}}{\sum_{t'} e^{\lambda \hat{U}_{t'}^a}} = \frac{e^{\lambda(w_1 x_t + w_2 R_t^a + w_3 P_t^a)}}{\sum_{t'} e^{\lambda(w_1 x_{t'} + w_2 R_{t'}^a + w_3 P_{t'}^a)}} \tag{5.2}$$

The problem of finding the optimal strategy for the defender can therefore be formulated as:

$$\max_x \sum_{t=1}^{T} \frac{e^{\lambda(w_1 x_t + w_2 R_t^a + w_3 P_t^a)}}{\sum_{t'} e^{\lambda(w_1 x_{t'} + w_2 R_{t'}^a + w_3 P_{t'}^a)}} (x_t R_t^d + (1 - x_t) P_t^d)$$
$$\text{s.t.} \sum_{t=1}^{T} x_t \leq K, 0 \leq x_t \leq 1 \tag{5.3}$$

Here, the objective is to maximize the defender's expected value given that the adversary chooses to attack each target with a probability according to the SUQR model. Constraint (5.3) ensures that the coverage probabilities on all the targets satisfy the resource constraint. Given that this optimization problem is similar to BRQR we use the same approach as BRQR to solve it. We refer the resulting algorithm as SU-BRQR.

5.1.1 Learning SUQR Parameters

Without loss of generality, we set $\lambda = 1$. We employ Maximum Likelihood Estimation (MLE) to learn the parameters (w_1, w_2, w_3). Given the defender strategy $\mathbf{x}$ and N samples of the players' choices, the log-likelihood of (w_1, w_2, w_3) is given by:

$$logL(w_1, w_2, w_3|x) = \sum_{j=1}^{N} log[q_{t_j}(w_1, w_2, w_3)]$$

where t_j is the target that is chosen in sample j and $q_{t_j}(w_1, w_2, w_3)$ is the probability that the adversary chooses the target t_j given the parameters (w_1, w_2, w_3). Let N_t be the number of subjects attacking target t. Then we have:

$$logL(w_1, w_2, w_3|x) = \sum_{t=1}^{T} N_t log[q_t(w_1, w_2, w_3)]$$

Combining with equation (2),

$$logL(w_1, w_2, w_3|x) = w_1(\sum_{t=1}^{T} N_t x_t) + w_2(\sum_{t=1}^{T} N_t R_t^a)$$
$$+w_3(\sum_{t=1}^{T} N_t P_t^a) - Nlog(\sum_{t=1}^{T} e^{w_1 x_t + w_2 R_t^a + w3 P_t^a})$$

$logL(w_1, w_2, w_3|x)$ can be shown to be a concave function: we can show that the Hessian matrix of $logL(w_1, w_2, w_3|x)$ is negative semi-definite. Thus, this function has an unique local maximum point and we can hence use a convex optimization solver to compute the optimal weights (w_1, w_2, w_3), e.g., *fmincon* in Matlab.

5.1.2 Prediction Accuracy of SUQR model

As in some real-world security environments, we would want to learn parameters of our SUQR model based on limited data. To that end, we used the data of 5 payoff structures and 2 algorithms MATCH and BRQR (10 games in total) from [Pita et al., 2012] to learn the parameters of the new SU function and the alternatives. In total, 33 human subjects played these 10 games using the setting of 8-targets and 3-guards from our on-line game. The parameters that we learnt are: (w_1, w_2, w_3)=$(-9.85, 0.37, 0.15)$ for the 3-parameter SU function; and $(w_1, w_2, w_3, w_4, w_5)$ = $(-8.23, 0.28, 0.12, 0.07, 0.09)$ for the 5-parameter function.

Table 5.1: Prediction Accuracy

QR	3-parameter SUQR	5-parameter SUQR
8%	51%	44%

We ran a Pearson's chi-squared goodness of fit test [Greenwood and Nikulin, 1996] in all the 100 payoff structures in [Pita et al., 2012] to evaluate the prediction accuracy of the two proposed models as well as the classic QR model. The test examines whether the predicted distribution of

the players' choices fits the observation. We set $\lambda = .76$ for QR model, the same as what was learned in [Yang et al., 2011]. The percentages of the payoff structures that fit the predictions of the three models (with statistical significance level of $\alpha = 0.05$) are displayed in Table 5.1. The table clearly shows that the new SUQR model (with the SU function in Equation (5.1)) predicts the human behavior more accurately than the classic QR model. In addition, even with more parameters, the prediction accuracy of the 5-parameter SUQR model does not improve. Given this result, and our 3-parameter model demonstrated superiority (as we will show in the Experiments section), we leave efforts to further improve the SUQR model for future work.

5.2 Improving MATCH

Since SUQR better predicts the distribution of the subject's choices than the classic QR, and as shown later, SU-BRQR outperforms MATCH, it is natural to investigate the integration of the subjective utility function into MATCH. In particular, we replace the expected value of the

adversary with subjective utility function. Therefore, the adversary's loss caused by his deviation from the optimal solution is measured with regard to the subjective utility function.

$$\max_{x,h,\eta,\gamma} \quad \gamma \tag{5.4}$$

$$\text{s.t.} \sum_{t \in T} x_t \leq K, 0 \leq x_t \leq 1, \qquad \forall t \tag{5.5}$$

$$\sum_{t \in T} h_t = 1, h_t \in \{0, 1\}, \qquad \forall t \tag{5.6}$$

$$0 \leq \eta - (w_1 x_t + w_2 R_t^a + w_3 P_t^a) \leq M(1 - h_t) \tag{5.7}$$

$$\gamma - (x_t R_t^d + (1 - x_t) P_t^d) \leq M(1 - h_t) \tag{5.8}$$

$$\gamma - (x_t R_t^d + (1 - x_t) P_t^d) \leq$$

$$\beta \cdot (\eta - (w_1 x_t + w_2 R_t^a + w_3 P_t^a)), \qquad \forall t \tag{5.9}$$

We refer to this modified version as SU-MATCH, which is shown in Equation (5.4)-(5.9) where h_t represents the adversary's target choice, η represents the maximum subjective utility for the adversary, γ represents the expected value for the defender if the adversary responds optimally and M is a large constant.

Constraint (5.7) finds the optimal strategy (target) for the adversary. In constraint (5.8), the defender's expected value is computed when the attacker chooses his optimal strategy. The key idea of SU-MATCH is in constraint (5.9). It guarantees that the loss of the defender's expected value caused by adversary's deviation is no more than a factor of β times the loss of the adversary's subjective utility.

5.2.1 Selecting β for MATCH:

In MATCH, the parameter β is the key that decides how much the defender is willing to lose if the adversary deviates from his optimal strategy. Pita et al. set β to 1.0, leaving its optimization for future work. In this section, we propose a method to estimate β based on the SUQR model.

---

1 Initialize $\gamma^* \leftarrow -\infty$;
2 **for** *i = 1 to N* **do**
3 $\quad \beta \leftarrow$ *Sample*$([0, \texttt{MaxBeta}], i)$, $x \leftarrow$ MATCH(β);
4 $\quad \gamma \leftarrow \sum_t q_t U_t^d$;
5 $\quad$ **if** $\gamma \geq \gamma^*$ **then**
6 $\quad\quad \gamma^* \leftarrow \gamma, \beta^* \leftarrow \beta$;
7 $\quad$ **end**
8 **end**
9 return (β^*, γ^*);

In this method, N values of β are uniformly sampled within the range (0, MaxBeta). For each sampled value of β, the optimal strategy x for the defender is computed using MATCH. Given this mixed strategy x, the defender's expected value, γ, is computed assuming that the adversary will respond stochastically according to the SUQR model. The β leading to the highest defender expected value is chosen. In practice, we set MaxBeta to 5, to provide an effective bound on the defender loss, given that penalties/rewards of both players range from -10 to 10; and N to 100, which gives a grid size of 0.05 for β for the range of $(0, 5)$. We refer to the algorithm with *carefully selected β as MATCHBeta*.

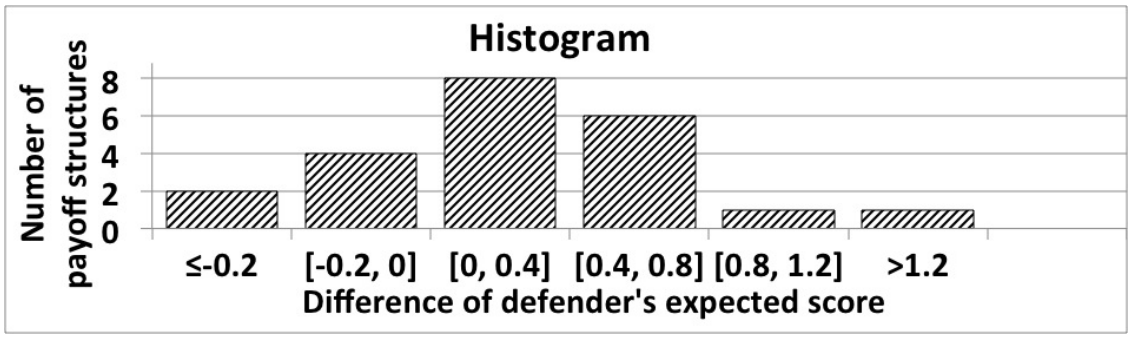
5.3 Experimental Results

In this section, I present the experiment results on comparing the MATCH algorithm (and its extended versions) with the SU-BRQR algorithm. We leave the comparison between SUQR model and QRRU model 4.1.3 for future work.

5.3.1 Results with AMT Workers, 8-target Games

Our first experiment compares SU-BRQR against MATCH and its improvements, in the setting where we learned the parameters of the SUQR model, i.e., the 8-target and 3-guard game with the AMT workers. In this 8-target game setting, for each game, our reported average is over at least 45 human subjects. The experiments were conducted on the AMT system. When two algorithms are compared, we ensured that identical human subjects played both on the same payoff structures. Participants were paid a base amount of US $\$1.00$. In addition, each participant was given a bonus based on their performance in the games to motivate them. Similar to [Pita et al., 2012]'s work, we ensured that players were not choosing targets arbitrarily by having each participant play two extra trivial games (i.e., games in which there is a target with the highest adversary reward and lowest adversary penalty and lowest defender coverage probability). Players' results were removed if they did not choose that target.

We generated the payoff structures based on covariance games in GAMUT [Nudelman et al., 2004]. In covariance games, we can adjust the covariance value $r \in [-1, 1]$ to control the correlation between rewards of players. We first generate 1000 payoff structures with r ranging from -1 to 0 by 0.1 increments (100 payoff structures per value of r). Then, for each of the 11 r values, we select 2 payoff structures ensuring that the strategies generated by each candidate algorithm (e.g., SU-BRQR and versions of MATCH) are not similar to each. One of these two has the maximum and the other has the median sum of 1-norm distances between defender strategies generated by each pair of the algorithms. This leads to a total of 22 payoff structures. By selecting the payoffs in this way, we explore payoff structures with different levels of the 1-norm distance between generated strategies so as to obtain accurate evaluations with regard to performance of the tested



	SU-BRQR	Draw	MATCH
$\alpha = .05$	13	8	1

Table 5.2: SU-BRQR vs MATCH, AMT workers, 8 targets

algorithms. We evaluate the statistical significance of our results using the bootstrap-t method [Wilcox, 2003].

5.3.2 SU-BRQR vs MATCH

This section evaluates the impact of the new subjective utility function via a head-to-head comparison between SU-BRQR and MATCH. In this initial test, the β parameter of MATCH was set to 1.0 as in [Pita et al., 2012]. Figure 5.3.2a shows all available comparison results for completeness. More specifically, we show the histogram of the difference between SU-BRQR and MATCH in the average defender expected reward over all the choices of the participants. The x-axis shows the range of this difference in each bin and the y-axis displays the number of payoff structures (out of 22) that belong to each bin. For example, in the third bin from the left, the average defender expected value achieved by SU-BRQR is larger than that achieved by MATCH, and the difference ranges from 0 to 0.4. There are 8 payoffs that fall into this category. Overall, SU-BRQR achieves a higher average expected defender reward than MATCH in the 16 out of the 22 payoff structures.

In Figure 5.3.2b, the second column shows the number of payoffs where SU-BRQR outperforms MATCH with statistical significance ($\alpha = .05$). The number of payoff structures where MATCH is better than SU-BRQR with statistical significance is shown in the fourth column. In the 22 payoff structures, SU-BRQR outperforms MATCH 13 times with statistical significance while MATCH defeats SU-BRQR only once; in the remaining 8 cases, no statistical significance is obtained either way. This result stands in stark contrast to [Pita et al., 2012]'s result and directly answers the question we posed at the beginning of this paper: there is indeed value to integrating models of human decision making in computing defender strategies in SSGs, but use of SUQR rather than traditional QR models is crucial. Furthermore, we ran the Person's chi-square goodness of fit test to evaluate the predication accuracy of the SUQR model and the traditional QR model, similar to that in Section 5.1. In all the 44 games (22 payoffs with 2 strategies for each), 20 games fit the predication of the SUQR model while only 7 games fit the prediction of the QR model.

5.3.3 SU-BRQR vs Improved MATCH

In Table 5.3, we compare MATCH and SU-BRQR against the three improved versions of MATCH: SU-MATCH, MATCHBeta, and SU-MATCHBeta (i.e., MATCH with both the subjective utility function and the selected β) when playing our 22 selected payoff structures. We report the results that hold with statistical significance ($\alpha = .05$). The first number in each cell in Table 5.3 shows the number of payoffs (out of 22) where the row algorithm obtains a higher average defender expected reward than the column algorithm; the second number shows where the column algorithm outperforms the row algorithm. For example, the second row and second column shows

Table 5.3: Performance comparison, $\alpha = .05$

	SU-MATCH	MATCHBeta	SU-MATCHBeta
MATCH	3, 11	1, 6	1, 8
SU-BRQR	8, 2	8, 2	5, 3

that MATCH outperforms SU-MATCH in 3 payoff structures with statistical significance while SU-MATCH defeats MATCH in 11.

Table 5.3 shows that the newer versions of MATCH achieve a significant improvement over MATCH. Additionally, SU-BRQR retains a significant advantage over both SU-MATCH and MATCHBeta. For example, SU-BRQR defeats SU-MATCH in 8 out of the 22 payoff structures with statistical significance, as shown in Table 5.3; in contrast, SU-MATCH is better than SU-BRQR only twice.

Although SU-BRQR in this case does not outperform SU-MATCHBeta to the extent it does against MATCH (i.e., SU-BRQR performs better than SU-MATCHBeta only 5 times with statistical significance while SU-MATCHBeta is better than SU-BRQR thrice (Table 5.3)), SU-BRQR remains the algorithm of choice for the following reasons: (a) SU-BRQR does perform better than SU-MATCHBeta in more cases with statistical significance; (b) selecting the β parameters in SU-MATCHBeta can be a significant computational overhead for large games given that it requires testing many values of β. Thus, we could just prefer SU-BRQR .

5.3.4 Results with New Experimental Scenarios

All previous experiments are based on the 8-target and 3-guards game, which were motivated by the LAX security scenario. In addition, the games have been played by AMT workers or college students. To evaluate the performance of the SUQR model in new scenarios, we introduce two new experimental settings: in one the experiments are conducted against a new type of human

	SU-BRQR	Draw	MATCH
$\alpha = .05$	6	13	3

Table 5.4: SU-BRQR vs MATCH, security experts

adversary, i.e., security intelligence experts; and in the other, we change the game to 24 targets and 9 guards.

5.3.4.1 Security Intelligence Experts, 8-target games

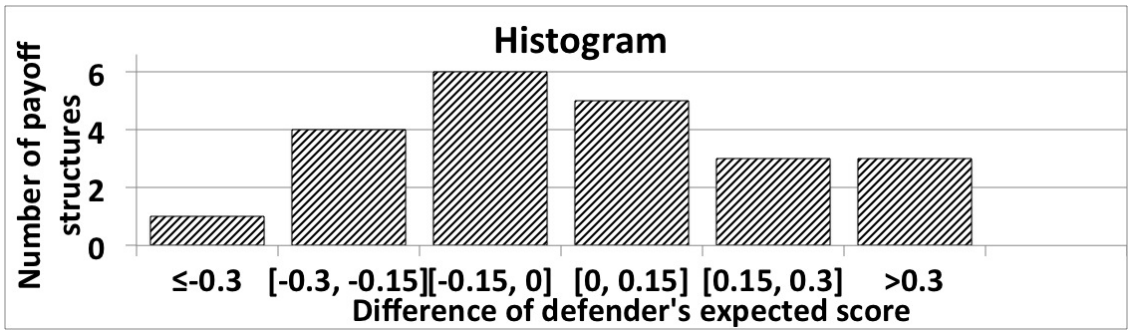
In this section, we evaluate our algorithm with security intelligence experts who serve in the best Israeli Intelligence Corps unit or are alumna of that unit. Our purpose is to examine whether SU-BRQR will work when we so radically change the subject population to security experts. We use the same 22 payoff structures and the same subjective utility function as in the previous experiment with AMT workers. Each result below is averaged over decisions of 27 experts.

5.3.4.2 SU-BRQR vs DOBSS

DOBSS performed poorly in 8-target games against AMT workers, as shown in Chapter 4. However, would DOBSS perform better in comparison to SU-BRQR against security experts? Our results show that SU-BRQR is better than DOBSS in all 22 tested payoff structures; 19 times with statistical significance. Thus, even these experts did not respond optimally (as anticipated by DOBSS) against the defender's strategies.

5.3.4.3 SU-BRQR vs MATCH

Figure 5.3.4.3a shows that SU-BRQR obtains a higher expected defender reward than MATCH in 11 payoff structures against our experts. Furthermore, SU-BRQR performs better than MATCH in 6 payoff structures with statistical significance while MATCH is better than SU-BRQR only



in 3 payoff structures with statistical significance (Figure 5.3.4.3b). These results still favor SU-BRQR over MATCH, although not as much as when playing against AMT workers (as in Figure 5.3.2).

Nonetheless, what is crucially shown in this section is that changing the subject population to security experts does not undermine SU-BRQR completely; in fact, despite using the data from AMT workers, SU-BRQR is still able to perform better than MATCH. We re-estimate the parameters (w_1, w_2, w_3) of the SU function using the data of experts. The result is: $w_1 = -11.0, w_2 = 0.54$, and $w_3 = 0.35$. This result shows that while the experts evaluated all the criteria differently from the AMT workers they gave the same importance level to the three parameters. Because of limited access to experts, we could not conduct experiments with these re-estimated parameters; we will show the impact of such re-estimation in our next experimental setting.

5.3.5 Bounded Rationality of Human Adversaries

We now compare the AMT workers and security experts using the traditional metric of "rationality level" of the QR model. To that end, we revert to the QR-model with the expected value function to measure how close these players are to perfect rationality. In particular, we use QR's λ parameter as a criterion to measure their rationality. We use all the data from AMT workers

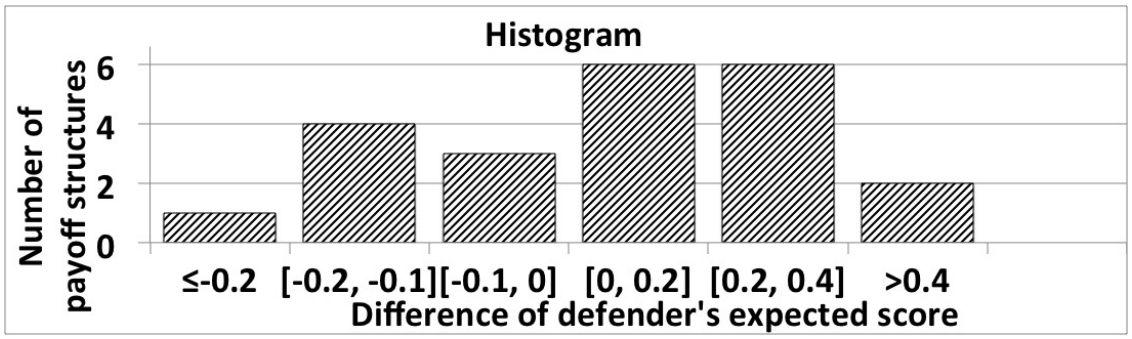
as well as experts on the chosen 22 games in previous experiments to learn the λ parameter. We get $\lambda = 0.77$ with AMT workers and $\lambda = 0.91$ with experts. This result implies that security intelligence experts tend to be more rational than AMT workers (the higher the λ, the closer the players are to perfect rationality). Indeed, in 34 of 44 games, experts obtains a higher expected value than AMT workers. Out of these, their expected value is higher than AMT workers 9 times while AMT workers' is higher only once with statistical significance ($\alpha = .05$). Nonetheless, the lambda for experts of 0.91 suggests that the experts do not play with perfect rationality (perfect rational $\lambda = \infty$).

5.3.6 AMT Workers, 24-target Games

In this section, we focus on examining the performance of the algorithms in large games, i.e., 24 targets and 9 defender resources. We expect that the human adversaries may change their behaviors because of tedious evaluation of risk and benefit for each target. Three algorithms were tested: SU-BRQR, MATCH, and DOBSS. We first run experiments with the new subjective utility function learned previously using the data of the 8-target game.

5.3.6.1 SU-BRQR vs MATCH with Parameters Learned from the 8-target Games

Figure 5.3.6.1a shows that SU-BRQR obtains a higher average defender expected value than MATCH in 14 out of 22 payoff structures while MATCH is better than SU-BRQR in 8 payoff structures. These averages are reported over 45 subjects. In addition, as can be seen in Figure 5.3.6.1b, SU-BRQR performs better than MATCH with statistical significance 8 times while MATCH outperforms SU-BRQR 3 times. While SU-BRQR does perform better than MATCH, its superiority over MATCH is not as much as it was in previous 8-target games.



	SU-BRQR	Draw	MATCH
$\alpha = .05$	8	11	3

Table 5.5: SU-BRQR vs MATCH, 24 targets, original

We can hypothesize based on these results that the learned parameters of the 8-target games do not predict human behaviors as well in the 24-target games. Therefore, we re-estimate the values of the parameters of the subjective utility function using the data of the previous experiment in the 24-target games. The training data contains 388 data points. This re-estimating results in $w_1 = -15.29, w_2 = .53, w_3 = .34$. Similar to the experts case, the weights in 24-target games are different from the ones in 8-target games but their order of importance is the same.

5.3.6.2 SU-BRQR vs DOBSS with Re-estimated Parameters

Since DOBSS has not been tested in the 24-target setting, we test it as a baseline. SU-BRQR outperforms DOBSS with statistical significance in all 22 tested payoff structures illustrating the superiority of SU-BRQR over a perfectly rational baseline.

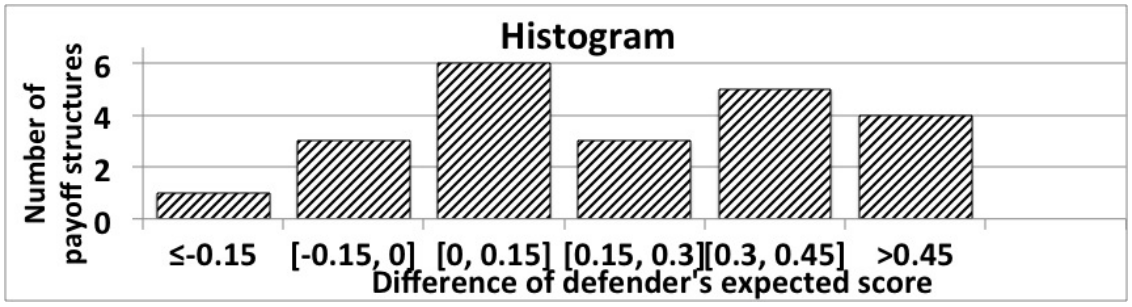
5.3.6.3 SU-BRQR vs MATCH with Re-estimated Parameters

In this experiment, we evaluate the impact of the new subjective utility function with the re-estimated parameters on the performance of SU-BRQR in comparison with MATCH.

	SU-BRQR	Draw	MATCH
$\alpha = .05$	11	10	1

Table 5.6: SU-BRQR vs MATCH, 24 targets, re-estimated

Figure 5.3.6.3a shows that SU-BRQR outperforms MATCH in 18 payoff structures while MATCH wins SU-BRQR in only 4 payoff structures. Moreover, it can be seen in Figure 5.3.6.3b that SU-BRQR defeats MATCH with statistic significance 11 times while MATCH defeats SU-BRQR only once with statistical significance. In other words, the new weights of the subjective utility function indeed help improve the performance of SU-BRQR . This result demonstrates that a more accurate SU function can help improve SU-BRQR's performance.



Chapter 6: Modeling Human Adversaries in Network Security Games

In this chapter, I initiate the study of human behavior models of adversaries in network security games, as well as the problem of designing defender strategies against such human adversaries. Many real-world security domains have structure that is naturally modeled as graphs. For example, in response to the devastating terrorist attacks in 2008 [Chandran and Beitchman, 2008], Mumbai police deployed randomized checkpoints as one countermeasure to prevent future attacks ([Ali, 2009]). This can be modeled as a network security game ([Washburn and Wood, 1995; Tsai et al., 2010; Jain et al., 2011a]), a Stackelberg game on a graph with intersections as nodes and roads as edges, where certain nodes are targets for attacks. The defender (as the leader) can schedule randomized checkpoints on edges of the graph. The attacker (as the follower) chooses a path on the graph ending at one of the targets.

A common assumption of these previous studies is that the attacker is perfectly rational (i.e. chooses a strategy that maximizes their expected utility). This is a reasonable proxy for the worst case of a highly intelligent attacker, but it can lead to a defense strategy that is not robust against attackers using different decision procedures, and it fails to exploit known weaknesses in the decision-making of human attackers. In previous chapters, we have considered security domains

where the human adversaries choose from a set of given targets, in a network security game the human attacker faces a more complex decision: that of choosing a path in a graph. On one hand, the rationality assumption is even more problematic here; on the other hand, the existing behavior models do not explicitly consider the specific graphical structure of this domain. For modeling human path planning in continuous terrains, Burgess and Darken ([Burgess and Darken, 2004]) proposed a fluid-simulation based model; however, their model is less applicable to our domain in which the choices are discrete.

In this chapter, I present the first systematic study of human behavior models applied to network security games. After formerly defining the problem in Section 6.1, I consider two behavior models for attackers in Section 6.2. First, I adapt the quantal response model to network security games. The second model, which I call quantal response with heuristics , is motivated by studies showing that humans rely on heuristics to address complex decision problems (e.g., [Gigerenzer et al., 1999]). Then in Section 6.3, I describe how the model parameters are estimated using data collected through a web-based game that I develop to simulate the decision tasks faced by the attacker. It then follows by Section 6.4, where I explain how to compute defender strategies that optimize defender utility against each of these behavior models of the attackers. Finally, in Section 6.3, I compare the performance of these strategies in a subsequent set of experiments on Amazon Mechanical Turk.

6.1 Problem Definition

We model a network security domain, similar to that introduced by Tsai *et al* [Tsai et al., 2010]. We use the following notation to describe the game, which are also listed in Table 9.1. The game

is played on a graph $G = (V, E)$. The attacker starts at one of the source nodes $s \in S \subset V$ and travels along a path chosen by him to get to one of the target nodes $t \in T \subset V$. The attacker's set of pure strategies $\mathcal{A}$ then consists of all the possible paths from some $s \in S$ to some $t \in T$, which we denote $A_1, \ldots, A_{|\mathcal{A}|} \subset E$. Meanwhile, the defender tries to catch the attacker by setting up check points on the passing edges before the attacker reaches the target. Let M be the total number of security resources, meaning the defender could then set up at most M simultaneous check points in the network. Thus the set of defender's pure strategies $\mathcal{D}$ consists of all subsets of E with at most M elements, which we denote $D_1, \ldots, D_{|\mathcal{D}|}$. If the attacker chooses a path which has at leat one edge covered by the defender, then the attacker gets caught and receives a penalty, and the defender receives a reward for catching the attacker; otherwise, the attacker receives a reward for successfully attacking the target and the defender receives a penalty. Formally, assuming the defender plays an allocation D_i, and the attacker chooses a path A_j, the attacker succeeds if and only if $D_i \cap A_j = \emptyset$.

The game was assumed to be zero-sum in earlier work [Tsai et al., 2010; Jain et al., 2011a]. In this paper, we relax this assumption to consider a more general class of games. Specifically, successful attacks might lead to different rewards to the attackers since different targets might be of different values to the attackers. Meanwhile, catching the attacker on different paths might give different rewards to the defender. We use R_j^a to denote the rewards received by attacker for a successful attack through path A_j, and P_j^d to denote the penalty received by the defender. If the attacker gets caught on path A_j, we denote his penalty by P_j^a and the reward received by the defender by R_j^d. Furthermore, we make the natural assumption that $R_i^a > P_i^a$ and $R_i^d > P_i^d$, $\forall i \in \{1, \ldots, |\mathcal{A}|\}$. Taking everything together, we define a *network security game* Γ as the tuple $(G, S, T, M, \{R_i^d\}, \{P_i^d\}, \{R_i^a\}, \{P_i^a\})$.

Table 6.1: Notations used in this paper

(V, E)	Network game graph
M	Total number of defender resources
$\mathcal{A}$	Set of attacker paths, $\mathcal{A} = \{A_i\}$
A_i	i^{th} attacker path
R_i^a	Reward for attacker for a successful attack through path A_i
P_i^a	Penalty for attacker if he gets caught on path A_i
R_i^d	Reward for defender for catching attacker on path A_i
P_i^d	Penalty for defender for a successful attack through path A_i
$\mathcal{D}$	Set of defender allocations (strategies), $\mathcal{D} = \{D_j\}$
D_j	j^{th} defender allocation
Γ	A network security game
x_e	Probability that edge e will be covered by a resource

The attacker conducts surveillance to learn about the defender's strategy, so it is important for the defender to randomize her strategy to avoid exploitable patterns. In other words, the defender has to commit to a distribution over her pure strategies. We use x_e to denote the probability that an edge $e \in E$ will be covered by the defender and $\boldsymbol{x} = \langle x_e, \forall e \in E \rangle$ to denote the vector of marginal probabilities of covering each of the edges in the graph. In general, if the attacker chooses path A_i the probability that he will be captured (denoted p_i) is the probability that at least one edge on the path A_i is covered by the defender, which is not completely specified by the marginals $\boldsymbol{x}$. Tsai *et al* [Tsai et al., 2010] showed that given $\boldsymbol{x}$, the sum of marginals on the edges of the path $\sum_{e \in A_i} x_e$ is an upper bound of p_i, and this upper bound can be reached if the defender can ensure that in each pure strategy D_j played with positive probability, only one edge on the path A_i is covered. Tsai *et al* [Tsai et al., 2010] proposed algorithms that sample defender pure strategies from $\boldsymbol{x}$, however such techniques are not guaranteed to reach this upper bound in all cases. In this paper, we make the simplifying assumption that the total amount of defender resources M is equal to 1, which is consistent with our focus on small graphs. Then since at most

one edge of the graph will be covered in any pure strategy D_j, we have $p_i \equiv p_i(\boldsymbol{x}) = \sum_{e \in A_i} x_e$ for all i. Then we can write the expected utility of the defender if attacker chooses path A_i as

$$U_i^d(\boldsymbol{x}; \Gamma) = p_i(\boldsymbol{x}) R_i^d + (1 - p_i(\boldsymbol{x})) P_i^d \tag{6.1}$$

and the expected utility for the attacker if he chooses A_i as

$$U_i^a(\boldsymbol{x}; \Gamma) = (1 - p_i(\boldsymbol{x})) R_i^a + p_i(\boldsymbol{x}) P_i^a \tag{6.2}$$

Let $q_i(\boldsymbol{x}; \Gamma)$ denote the probability that attacker chooses path A_i, given the defender's marginal coverage on all the edges $\boldsymbol{x}$. The optimal strategy for the defender is to maximize the average expected utility:

$$\max_{\boldsymbol{x}} \sum_{A_i \in \mathcal{A}} q_i(\boldsymbol{x}; \Gamma) U_i^d(\boldsymbol{x}; \Gamma) \tag{6.3}$$

It is thus important for the defender to accurately model the attacker's response to her strategy, i.e., $q_i(\boldsymbol{x}; \Gamma)$ for all i.

We assume that the attacker can observe M (which is equal to 1), as well as the the defender's marginal coverage on all the edges $\boldsymbol{x}$. A fully rational attacker would be able to deduce that $p_i = \sum_{e \in A_i} x_e$ for all i and choose a path that maximizes his expected utility: $i^* = \arg\max_i U_i^a(\boldsymbol{x}; \Gamma)$. However in real-world security problems, we are facing human attackers who may not respond optimally. The goal of this paper is to explore models that can better predict the behavior of human attackers.

6.2 Adversary Model

In this section, we propose several models of how a human attacker responds to the defender's strategy.

6.2.1 Basic Quantal Response Model

In our first model the attacker's mixed strategy is a quantal response (QR) to the defender's strategy. Under this QR model, given a graph game Γ and a defender's strategy $\boldsymbol{x}$, the probability that the adversary is going to choose path A_i is

$$\texttt{QR}: \quad q_i(\lambda \mid \boldsymbol{x}; \Gamma) = \frac{e^{\lambda U_i^a(\boldsymbol{x};\Gamma)}}{\sum_{A_k \in \mathcal{A}} e^{\lambda U_k^a(\boldsymbol{x};\Gamma)}} \tag{6.4}$$

where $\lambda > 0$ is the parameter of the quantal response model, which represents the error level of adversary's quantal response. When $\lambda = 0$, the adversary chooses each path with equal probability; when $\lambda = \infty$, the adversary becomes fully rational and only selects the paths which give him the maximum expected utility. It is shown in many empirical studies that λ usually takes a positive finite value.

6.2.2 Quantal Response with Heuristics

In a network security game Γ, in order to evaluate the expected utility of a path A_i, $U_i^a(\boldsymbol{x}; \Gamma)$, the attacker has to compute p_i, which requires reasoning about a sequence of random events, i.e., whether or not each edge on the path will be covered by the defender. Even in our simplified games in which $M = 1$ and thus a perfectly-rational attacker can compute p_i as the sum $\sum_{e \in A_i} x_e$, Computing this probability can be more difficult for bounded-rational human attackers who might not know this formula. Instead, the adversary might use simple heuristics to evaluate the "utility" of each path.

We propose the following model of the attacker's behavior which we call Quantal Response with Heuristics (QRH):

$$\texttt{QRH}: \quad h_i(\mu \mid \boldsymbol{x}; \Gamma) = \frac{e^{\mu \cdot f_i(\boldsymbol{x})}}{\sum_{A_k \in \mathcal{A}} e^{\mu \cdot f_k(\boldsymbol{x})}} \tag{6.5}$$

Table 6.2: Lists of Path Features

$f_{i1}(\boldsymbol{x}) := \sum_{e \in E} A_{ie}$	Number of edges
$f_{i2}(\boldsymbol{x}) := \max_{e \in A_i} x_e$	Minimum edge coverage
$f_{i3}(\boldsymbol{x}) := \min_{e \in A_i} x_e$	Maximum edge coverage
$f_{i4}(\boldsymbol{x}) := \sum_{e \in A_i} x_e$	Summation of edge coverage
$f_{i5}(\boldsymbol{x}) := f_{i4}(\boldsymbol{x}) / f_{i1}((\boldsymbol{x}))$	Average edge coverage

where $\mu = \langle \mu_1, ..., \mu_m \rangle$ is a vector of coefficients of the model and given $\boldsymbol{x}$, $f_i(\boldsymbol{x}) = \langle f_{i1}(\boldsymbol{x}), .., f_{im}(\boldsymbol{x}) \rangle$ is a vector of m *features* for path A_i that influences the attacker's decision making.

We observe that under both QR and QRH models the attacker's mixed strategy belongs to the *exponential family* of distributions widely used in statistical learning. The form of the QRH model is more general than QR: it allows linear combinations of multiple features, and furthermore $f_{ij}(\boldsymbol{x})$ can be any function, including the attacker's expected utility $U_i^a(\boldsymbol{x}; \Gamma)$ used in the QR model. On the other hand, since our focus for the QRH model is on simple heuristics, we use a set of five features that are easy to compute for humans and thus could be used as basis for heuristics. These features are listed in Table 6.2.

6.3 Model Parameter Estimation

6.3.1 Data Collection

In order to estimate the values of the parameters of our models, we first need data on how humans behave when faced with the kind of decision tasks the attacker faces. We developed a web-based game which simulates the decision tasks faced by the attacker in network security games, and collected data on how human subjects play the game by posting the game as a Human Intelligent Task (HIT) on Amazon Mechanical Turk (AMT).[1]

[1] https://www.mturk.com

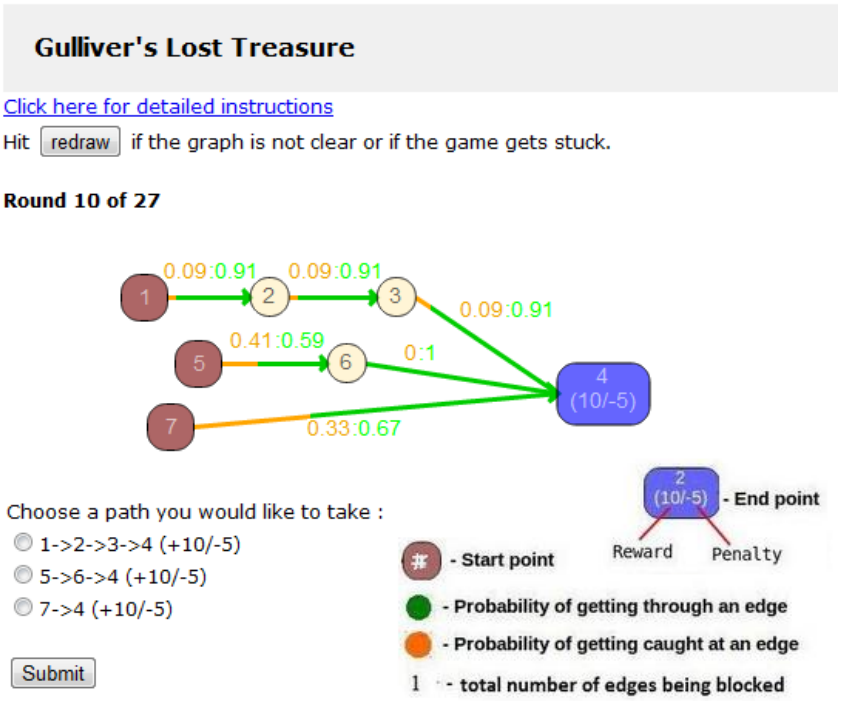
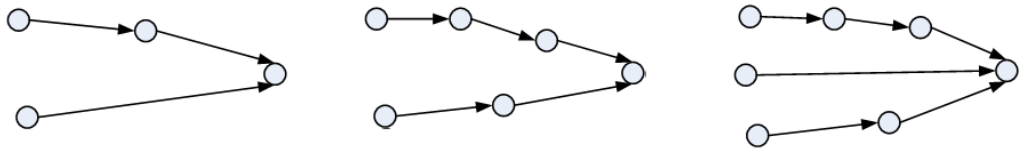


Figure 6.1: Game Interface (colored)

Figure 6.1 displays the interface of the game. Players were introduced to the game through a series of explanatory screens describing how the game is played. In the game, the web interface presents a graph to the subjects and specifies the source(starting) nodes and the target nodes in the graph. The subjects are asked to select a path from one of the source nodes to one of the target nodes. They are also told that the defender is trying to catch them by setting up checkpoints on the edges. The probability that there will be a check point on each edge is given to the subjects, as well as the reward for successfully getting through the path and the penalty for being caught by the defender. Thus each instance of this game can be specified by a network security game and a defender strategy. Formally, we define a *game sample* as $g = (\Gamma, \boldsymbol{x})$, where Γ is a network security game and $\boldsymbol{x}$ is a defender strategy. Each human subject plays multiple rounds in sequence, each corresponding to a different game sample. In each game round, after a subject selects a path in the



Graph 1 Graph 2 Graph 3

Figure 6.2: Graphs Tested in Data Collection

network, the edges that will be covered by the defender is sampled according to the probability shown in the figure. Subjects get a positive score if they successfully get through the path and a negative score if they select a path which has edges covered by the defender. In order to mitigate learning effects, subjects were not told of the result of each game round until they finish all game rounds. Each subject receives $0.5\$$ for participating in the experiments, and is paid $0.01\$$ bonus for each point they earn. In our experiments subjects earned $1.1\$$ bonus on average.

We conducted a first set of experiments on three simple graphs, shown in Figure 6.2. Since the purpose of this set of experiments is to collect data to train our models, we want to use a wide variety of defender strategies. We first randomly generated 1000 different defender strategies for each graph. We then used k-means clustering to classify these random strategies into K clusters. The centers of the clusters are selected as the representative strategies and used in the experiments. We selected 10 strategies for Graph 1, 10 strategies for Graph 2 and 20 strategies for Graph 3; details on the strategies can be found at an online appendix.[2] In total, we tested 40 different game samples, each of which are played by 40 different subjects.

[2]http://anon-aamas2012-paper826.webs.com/

6.3.2 Training the QR Model

We first train the basic quantal response model, `QR`, using the data collected in the experiment described in Section 6.3.1. We use Maximum Likelihood Estimation (MLE) to tune the parameter λ.

Given the choices of N subjects, with $\tau(n)$ denoting the path chosen by player n, the log-likelihood of λ on game sample g is

$$\log L_{QR}(\lambda \mid g) = \sum_{n=1..N} \log q_{\tau(n)}(\lambda \mid \boldsymbol{x}; \Gamma)$$

Let N_i be the number of subjects attacking target i. Then,

$$\log L_{QR}(\lambda \mid g) = \sum_{A_i \in \mathcal{A}} N_i \log q_i(\lambda \mid \boldsymbol{x}; \Gamma)$$

Combining with Equation (6.4),

$$\log L_{QR}(\lambda \mid g) = \lambda \sum_{A_i \in \mathcal{A}} N_i U_i^a(\boldsymbol{x}) - N \log(\sum_{A_i \in \mathcal{A}} e^{\lambda U_i^a(\boldsymbol{x})}) \tag{6.6}$$

We train the model by maximizing the total log-likelihood of all the 40 game samples

$$\max_{\lambda} \sum_{g \in \mathcal{S}} \log L_{QR}(\lambda \mid g) \tag{6.7}$$

where $\mathcal{S}$ denotes the set of all 40 game samples. It is relatively straightforward to verify that the second order derivative of $\log L_{QR}(\lambda \mid g)$ is always nonpositive. Thus $\log L_{QR}(\lambda \mid g)$ is a concave function in λ for all g. Therefore, the total log-likelihood of Equation 6.7 is concave and we can apply any local optimization solver (we used Matlab's `fmincon` solver). The maximum-likelihood estimate of λ based on the data is 0.34.

6.3.3 Training the QRH Model

In training the `QRH` model, we need to first decide which subset of the 5 features from Table 6.2 to use in the model, and then train the model for the selected features. Although in general the more features we select the better the fit will be, taking the set of all features can result in over-fitting. This *feature selection* problem is well-studied in statistics and machine learning, and techniques such as L1-regularized regression methods were proposed to introduce bias towards smaller sets of features. In this paper we apply a simple form of bias: we consider only subsets of features of sizes 1 and 2. We then select the top-performing subsets of size 1 and the top-performing subsets of size 2. Specifically, for each $L \in \{1, 2\}$, we do the following:

1. For each of the $\binom{5}{L}$ possible subsets of size L, we train a QRH model using this subset of features using MLE;

2. We compare the models using 2-fold cross validation, and pick the top two feature combinations.

Since we are only selecting from 5 features, we only have to evaluate a small number of models. In future work we plan to explore more sophisticated feature-selection techniques, which would allow us to select from a large set of possible features.

In order to apply 2-fold cross validation, we first randomly divided all the 40 game samples into two equal-sized sets, $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$. We conducted two rounds of training, one using $\boldsymbol{S}_1$ and the other using $\boldsymbol{S}_2$. In each round of training, the model is trained by maximizing the total log-likelihood of the game samples in the training set:

$$\max_{\mu} \sum_{g \in \mathcal{S}_{train}} \log L_{QRH}(\mu \mid g), \tag{6.8}$$

where $\mathcal{S}_{train} \in \{\mathcal{S}_1, \mathcal{S}_2\}$ is the training set, and $\log L_{QRH}(\mu \mid g)$ is the log-likelihood of QRH model of game sample g, derived similarly as $\log L_{QR}(\lambda \mid g)$:

$$\log L_{QRH}(\mu \mid g) = \mu \cdot (\sum_{A_i \in \mathcal{A}} N_i f_i(\boldsymbol{x})) - N \log(\sum_{A_i \in \mathcal{A}} e^{\mu \cdot f_i(\boldsymbol{x})}). \tag{6.9}$$

We can show that $\log L_{QRH}(\mu \mid g)$ is a concave function in μ, since the Hessian matrix is negative definite. Therefore, it can be solved use any local optimization solver.

features	Train on $\boldsymbol{S}_1$		Train on $\boldsymbol{S}_2$		Total
	training	testing	training	testing	testing
1	-707.2	-672	-636.8	-744.4	-1416.4
2	-693.6	-666	-658	-702	-1368
3	-636	-580.4	-573.6	-642.8	***-1223.2***
4	-677.2	-723.6	-710	-690.8	-1414.4
5	-667.6	-618.4	-606.4	-680.4	***-1298.8***
U_i^a	-645.6	-689.6	-682.8	-652.4	-1342

Table 6.3: Fit ($\log L$) of model `QRH` using single feature

features	Train on $\boldsymbol{S}_1$		Train on $\boldsymbol{S}_2$		Total
	training	testing	training	testing	testing
(1,2)	-693.2	-672	-635.2	-734.4	-1406.4
(1,3)	-630.4	-594.4	-570	-657.2	-1251.6
(1,4)	-603.6	-602	-573.6	-636	***-1238***
(1,5)	-648.8	-638.4	-606	-684.4	-1322.8
(2,3)	-636	-582.8	-572.4	-656.8	-1239.6
(2,4)	-631.6	-655.2	-638	-649.6	-1304.8
(2,5)	-643.6	-581.2	-566.4	-660.8	-1242
(3,4)	-616	-601.6	-573.6	-644	-1245.6
(3,5)	-636	-581.2	-571.2	-646.8	***-1228***
(4,5)	-610.4	-615.6	-592.4	-635.6	-1251.2

Table 6.4: Fit of model `QRH` using two features

Given a combination of the features f_i, let μ^1 and μ^2 be the training results on $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$, respectively. We measure the model fit of f_i as the sum of the log-likelihoods of $\boldsymbol{S}_2$ under the model for μ^1 and $\boldsymbol{S}_1$ under the model for μ^2:

$$\texttt{Fit}(f_i) = \sum_{g \in \mathcal{S}_2} \log L_{QRH}(\mu^1 \mid g) + \sum_{g \in \mathcal{S}_2} \log L_{QRH}(\mu^2 \mid g) \tag{6.10}$$

Table 6.3 displays the fit results for single features. For comparison, we also conduct the MLE training with 2-fold cross validation for the QR model and list the fitting result on the last row in Table 6.3. Over all, feature 3 (maximum edge coverage) achieves the best fitting performance, which is also better than the QR model. Additionally, feature 5 (average edge coverage) also achieves better fitting performance than the QR model. Table 6.4 displays the fit result with two features. The best two combinations are $(1, 4)$ and $(3, 4)$.

features	parameter value
3	-9.95
5	-6.26
(1,4)	(1.04, -10.60)
(3,4)	(-9.67, -1.95)

Table 6.5: Number of strategies tested

Based on the 2-fold cross validation results, we selected four candidate feature combinations for the QRH model: feature 3, feature 5, feature 1 + feature 4, feature 3 + feature 4. We then tuned the model parameters for these candidates by training on the whole data set $\boldsymbol{S}$. The final values for the parameters are listed in Table 6.5.

6.4 Computing Defender Resource Allocation Strategy

In this section, we describe how we compute optimal defender strategies against different models of attackers.

6.4.1 Best Response to QR model

Given a QR model of the adversary, the defender's expected utility by playing strategy $\boldsymbol{x}$ in a network security game Γ is:

$$\sum_{A_i \in \mathcal{A}} q_i(\lambda \mid \boldsymbol{x}; \Gamma) U_i^d(\boldsymbol{x}; \Gamma). \tag{6.11}$$

Combining with Equation (6.1) we have the following optimization problem to compute the defender's optimal strategy against a QR model of the adversary:

$$\max_{\boldsymbol{x},p} \frac{\sum_{A_i \in \mathcal{A}} e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)p_i}((R_i^d - P_i^d)p_i + P_i^d)}{\sum_{A_i \in \mathcal{A}} e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)p_i}} \tag{6.12}$$

$$\text{s.t.} \sum_{e \in E} x_e \leq M \tag{6.13}$$

$$p_i = \sum_{e \in A_i} x_e, \quad \forall A_i \in \mathcal{A} \tag{6.14}$$

$$0 \leq x_e \leq 1, \quad \forall e \in E \tag{6.15}$$

where $\lambda = 0.34$ as learned from the data. Since the defender is assumed to have only one resource, Constraint (6.14) ensures that p_i is the probability that path A_i will be covered by the defender. The objective function, Equation (6.12), is a nonlinear fractional function, thus is not guaranteed to be concave. We use a heuristic algorithm based on local optimization with random restarts, described in Algorithm 4. The algorithm generates a new starting point in each iteration and (at Line 5) calls `FindLocalMaximum` to find a locally optimal solution of (6.12). The best local optimal solution is returned in the end. We used Matlab's `fmincon` as the local optimizer.

Algorithm 3: Local Search with Random Multi-Restart

Input: $IterN$;
$opt_g \leftarrow -\infty$;
for $i \leftarrow 1, ..., IterN$ **do**
    $x_0 \leftarrow$ randomly generated feasible starting point;
    $(opt_l, x^*) \leftarrow \texttt{FindLocalMaximum}(x_0)$;
    if $opt_l > opt_g$ **then**
        $opt_g \leftarrow opt_l$, $x_{opt} \leftarrow x^*$
    end
end
return opt_g, x_{opt};

6.4.2 Best Response to QRH model

In this section, we explain our approach for computing an optimal defender strategy against a QRH model given any combination of features $f_i(\boldsymbol{x})$ and the corresponding feature coefficients μ.

Given a network security game Γ and the defender's strategy $\boldsymbol{x}$, the probability that the attacker will select path A_i is $h_i(\mu \mid \boldsymbol{x}; \Gamma)$ as: defined by Equation 6.5. Then the defender's expected utility can be written as

$$\sum_{A_i 1 \mathcal{A}} h_i(\mu \mid \boldsymbol{x}; \Gamma) U_i^d(\boldsymbol{x}; \Gamma).$$

Therefore we can formulate the defender's optimal strategy as the solution of the following optimization problem:

$$\max_{\boldsymbol{x}, p} \frac{\sum_{A_i \in \mathcal{A}} e^{\mu \cdot f_i(\boldsymbol{x})} ((R_i^d - P_i^d) p_i + P_i^d)}{\sum_{A_i \in \mathcal{A}} e^{\mu \cdot f_i(\boldsymbol{x})}} \tag{6.16}$$

$$\text{s.t.} \sum_{e \in E} x_e \leq M \tag{6.17}$$

$$p_i = \sum_{e \in A_i} x_e, \quad \forall A_i \in \mathcal{A} \tag{6.18}$$

$$0 \leq x_e \leq 1, \quad \forall e \in E \tag{6.19}$$

where $f_i(\boldsymbol{x})$ is a subset of the features described in Table 6.2. Again, the objective function (6.16) is a nonlinear fractional function, so is not guaranteed to be concave. Nevertheless we can apply Algorithm 4, with `FindLocalMaximum` to find a locally optimal solution of (6.16).

6.5 Experiment Results

In this section, we evaluate the performance of different models in network security games. We use the same web-based game that we introduced in Section 6.3.1 to set up the experiments with human subjects. Different from the first set of experiments, where we intended to collect data on how humans play the game in order to train the model, the goal of this new set of experiments is to use the defender strategies computed from the different models to play against human subjects in order to compare the performance of these models.

6.5.1 Experiment Settings

Fig. 6.1 shows the interface of the web-based game we developed. We have provided details on the game rules in Section 6.3.1. We now focus on describing the game instances that are included in these experiments.

We tested eight different graph types, including the three graphs used in data collection that are displayed in Figure 6.2. The other five graphs are displayed in Figure 6.3. Among the eight graphs, we have four graphs with a single target (graph 1-4) and four graphs with multiple targets (graph 5-8). The models are trained using the data from single-target graphs, we are interested to see how they perform in multi-target graphs. For each graph type, we designed two different

Uniform	Defender covers each edge with equal probability
Maximin	Attacker always chooses the worst path for the defender
Rational	Attacker maximizes his expected utility
QR	quantal response ($\lambda = 0.34$)
QRH-1	QRH with maximum edge coverage ($\mu = -9.95$)
QRH-2	QRH with average edge coverage ($\mu = -6.26$)
QRH-3	QRH with number of edges and sum of edge coverage ($\mu = \langle 1.04, -10.60 \rangle$)
QRH-4	QRH with maximum edge coverage and sum of edge coverage ($\mu = \langle -9.67, -1.95 \rangle$)

Table 6.6: Attacker Models Tested Evaluated

sets of payoffs (i.e. the reward/penalty for the attacker and the defender on each path)[3]. Therefore, we have a total of $8 * 2 = 16$ security games in the experiments. For each of these games, we computed the defender strategies from eight different models. Table 6.6 lists the eight models. Therefore, for each game instance, we have eight different defender strategies. In total, we have $8 * 16 = 128$ different game samples (i.e., combinations of security games and defender strategies). Each of the game samples is played by 40 different subjects.

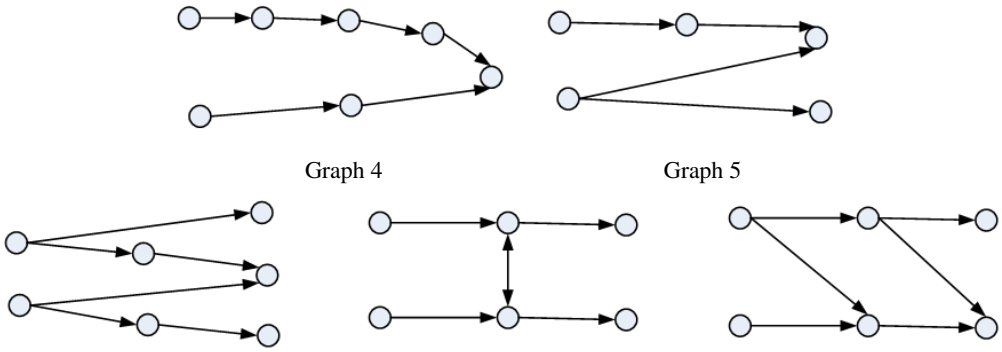


Figure 6.3: Graphs Tested in Evaluation Experiments

[3]The details of the payoffs can be found on the online appendix: http://anon-aamas2012-paper826.webs.com/

6.5.2 Experiment Results

We evaluate the performance of different defender strategies using the defender's expected utility. Given that a subject selects path A_i, the defender's expected utility is computed with Equation (6.1).

Average Performance: We first evaluated the average defender expected utility, $U^d_{exp}(\boldsymbol{x})$, of different defender strategies based on all 40 subjects choices:

$$U^d_{exp}(\boldsymbol{x}) = \frac{1}{40} \sum_{A_i \in \mathcal{A}} N_i U^d_i(\boldsymbol{x})$$

where N_i is the number of subjects that chose path A_i.

Figure 6.4 displays the average performance of the different models in all the single-target games on the left group of bars, and the average model performances in all the multi-target graphs on the right group of bars. In both cases, the QR model outperforms the three baseline models (Uniform, Maximin and Rational). Among the four QRH models, QRH-2 (i.e. using average edge coverage) and QRH-3 (i.e. using number of edges and sum of edge coverage) outperforms the three baseline models in both single-target graphs and multi-target graphs. There are two interesting observations from the figure.

- Between the QR model and the QRH models, we see that in the single-target graphs, none of the QRH models achieves better performance than the QR model; while in the mutli-target graphs, all four QRH models outperform QR. This is an unexpected result since the QRH models are trained on the single-target graph data and do not use features that come up in the multi-target graphs.

- The rational model did worse in the multi-target graphs than it did in the single-target graphs, as compared to the QR and QRH models. For the single-target graphs the average

defender expected utility achieved by the rational model was closer to that of the QR and QRH models, and it even outperformed two of the QRH models (QRH-1 and QRH-4). While in the multi-target graphs, the rational model was significantly outperformed by both QR and QRH models. This is also an surprising result, since the QR and QRH models are trained on the single-target graphs and are thus expected to perform better in the singe-target graphs.

A possible reason for the above two interesting observations is that as the graph becomes more complex (i.e. more targets and more paths), it becomes more difficult for humans to compute the actual expected utility of each path so they are more likely to rely on heuristics.

We also show the performance of different models in each graph type: Figures 6.5(a) shows the average defender expected utility achieved by the eight models in the four single-target graphs; and 6.5(b) displays the results in the four multi-target graphs. We can see from Figure 6.5(a) that the rational model was outperformed by the QRH-3 model in all of the four graphs; it was also outperformed by the QR model in 3 of the 4 graphs except for in graph 2 where the two models have roughly the same performance. In the multi-target graphs, Figure 6.5(b) shows that the rational model was outperformed by both the QR and QRH models in three of the four graphs, except for in graph 7 where all of the models have very similar performances.

Model Fitting Performance: We also evaluated the fitting performance of the five trained models. Table 6.7 reports the total log-likelihood of different models in the multi-target games and the single-target games.[4] It is clear that the four QRH models achieve much better fitting performance (i.e. higher log-likelihood) than the QR model. An interesting finding here is that better fit performance does not necessarily lead to higher defender expected utility. In particular,

[4]Detailed model fitting results can be found at the online appendix: http://anon-aamas2012-paper826.webs.com.

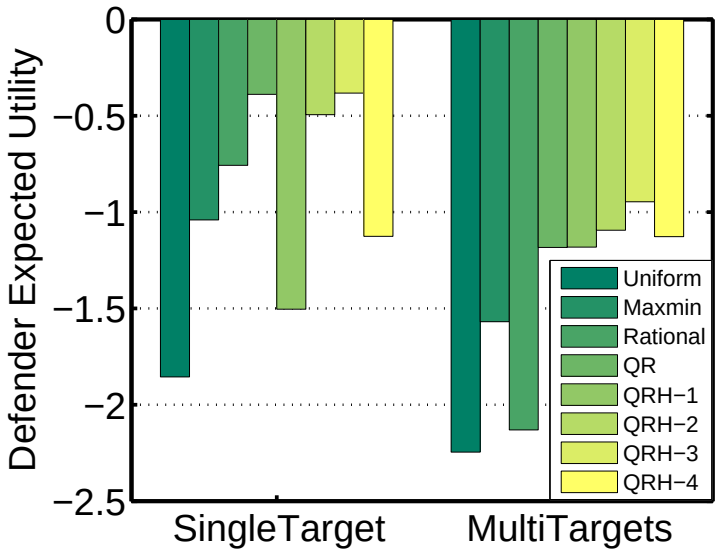
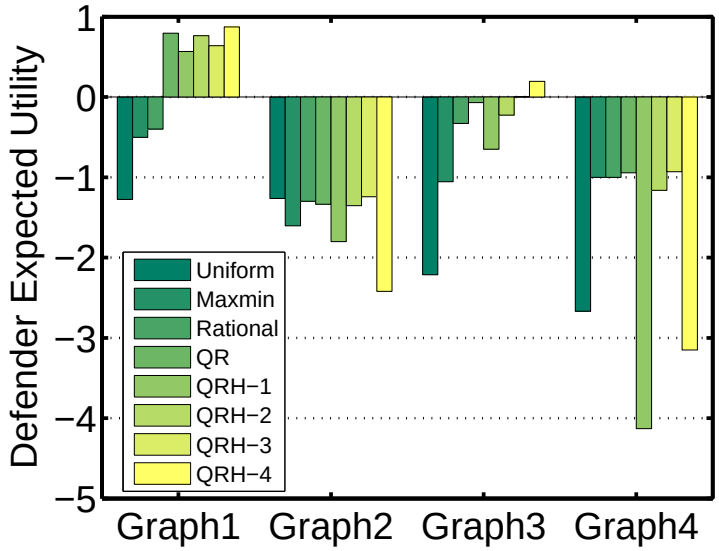


Figure 6.4: Average Defender Expected Utility

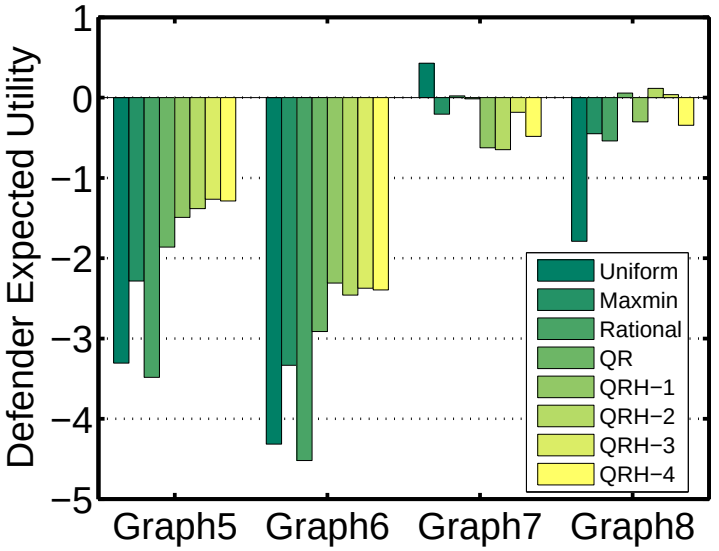
	Multi-Target (8 game instance)	Single-Target (8 game instance)	Total
QR	-397.09	-232.04	-629.13
QRH-1	-297.51	-208.27	-505.78
QRH-2	-349.60	-193.31	-542.90
QRH-3	-283.74	-168.43	-452.18
QRH-4	-279.43	-168.03	-447.46

Table 6.7: Fitting Performance: log-likelihood of different models

QRH-4 has the best fitting performance in single-target graphs, but the average defender expected utility achieved by QRH-4 was much worse than other models except for QRH-1, as shown in Figure 6.4.



(a) Single Target Graphs



(b) Multiple Target Graphs

Figure 6.5: Average Defender Expected Utility

Chapter 7: Computing Defender Optimal Strategy

In this chapter, I address the problem of computing optimal defender strategies in real-world security games against a quantal response model of attackers. The difficulties faced here include (1) solving a non-convex optimization problem efficiently for massive real-world security games; and (2) addressing constraints on assigning security resources, which adds to the complexity of computing the optimal defender strategy. I have introduced BRQR in Section 4.2 to compute defender optimal strategy against a quantal response. BRQR however was not guaranteed to converge to the optimal solution, as it used a nonlinear solver with multi-starts to obtain an efficient solution to a non-convex optimization problem. Furthermore, it does not consider resource assignment constraints that might be involved in my real-world security domains.

This chapter presents two new algorithms to address these difficulties. After formerly defining the problem in Section 7.1, Section 7.2 introduces the basic idea of using binary search to iteratively compute the defender optimal strategy against a quantal response model of the adversary. Section 7.3 then presents the GOSAQ algorithm which can compute the globally optimal defender strategy against a QR model of attackers when there are no resource constraints and gives an efficient heuristic otherwise. It then follows with Section 7.4, which provides an efficient approximation of the optimal defender strategy with or without resource constraints through

the algorithm PASAQ. Finally, Section 7.5 presents detailed experimental results showing the advantages of GOSAQ and PASAQ in solution quality over the benchmark algorithm (BRQR) and the efficiency of PASAQ.

7.1 Problem Definition

Assuming a QR-adversary, i.e. with a quantal response $\langle q_i, i \in \mathcal{T} \rangle$ to the defender's mixed strategy $\boldsymbol{x} = \langle x_i, i \in \mathcal{T} \rangle$. The value q_i is the probability that adversary attacks target i, computed as

$$q_i(\boldsymbol{x}) = \frac{e^{\lambda U_i^a(x_i)}}{\sum_{k \in \mathcal{T}} e^{\lambda U_k^a(x_k)}} \tag{7.1}$$

where $\lambda \geq 0$ is the parameter of the quantal response model, which represents the error level in adversary's quantal response. Simultaneously, the defender maximizes her utility (given her computer-aided decision making tool):

$$U^d(\boldsymbol{x}) = \sum_{i \in \mathcal{T}} q_i(\boldsymbol{x}) U_i^d(x_i)$$

Therefore, in domains without constraints on assigning the resources, the problem of computing the optimal defender strategy against a QR-adversary can be written in terms of marginals as:

$$\texttt{P1}: \begin{cases} \max\limits_{\boldsymbol{x}} \dfrac{\sum_{i \in \mathcal{T}} e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)x_i}((R_i^d - P_i^d)x_i + P_i^d)}{\sum_{i \in \mathcal{T}} e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)x_i}} \\ \text{s.t.} \sum\limits_{i \in \mathcal{T}} x_i \leq M \\ \quad 0 \leq x_i \leq 1, \quad \forall i \in \mathcal{T} \end{cases}$$

Problem P1 has a polyhedral feasible region and is a non-convex fractional objective function.

7.1.1 Resource Assignment Constraint

As I have shown in Section 2.1.3, there could be arbitrary constraints on assigning the security resources in real world security problems. A resource assignment constraint implies that the feasible assignment set $\mathcal{A}$ is restricted; not all combinatorial assignment of resources to targets are allowed. Hence, the marginals on targets, $\boldsymbol{x}$, are also restricted.

In order to compute the defender's optimal strategies against a QR-adversary in the presence of resource-assignment constraints, we need to solve `P2`. The constraints in `P1` are modified to enforce feasibility of the marginal coverage.

$$\texttt{P2}: \begin{cases} \max\limits_{\boldsymbol{x},a} \dfrac{\sum_{i\in\mathcal{T}} e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)x_i}((R_i^d - P_i^d)x_i + P_i^d)}{\sum_{i\in\mathcal{T}} e^{\lambda R_i^a} e^{-\lambda(R_i^a - P_i^a)x_i}} \\ \text{s.t.} \sum\limits_{i\in\mathcal{T}} x_i \le M \\ x_i = \sum\limits_{A_j\in\mathcal{A}} a_j A_{ij}, \quad \forall i \in \mathcal{T} \\ \sum\limits_{A_j\in\mathcal{A}} a_j = 1 \\ 0 \le a_j \le 1, \quad \forall A_j \in \mathcal{A} \end{cases}$$

7.2 Binary Search Method

We need to solve `P1` and `P2` to compute the optimal defender strategy, which requires optimally solving a non-convex problem which is in general an NP-hard problem [Vavasis, 1995]. In this section, we describe the basic structure of using a binary search method to solve the two problems. However, further efforts are required to convert this skeleton into actual efficiently runnable algorithms. We will fill in the additional details in the next two sections.

Table 7.1: Symbols for Targets in SSG

$\theta_i := e^{\lambda R_i^a} > 0$	$\beta_i := \lambda(R_i^a - P_i^a) > 0$	$\alpha_i := R_i^d - P_i^d > 0$

For notational simplicity, we first define the symbols $\forall\ i \in \mathcal{T}$ in Table 7.1. We then denote the numerator and denominator of the objective function in `P1` and `P2` by $N(\boldsymbol{x})$ and $D(\boldsymbol{x})$:

- $N(\boldsymbol{x}) = \sum_{i\in\mathcal{T}} \theta_i \alpha_i x_i e^{-\beta_i x_i} + \sum_{i\in\mathcal{T}} \theta_i P_i^d e^{-\beta_i x_i}$
- $D(\boldsymbol{x}) = \sum_{i\in\mathcal{T}} \theta_i e^{-\beta_i x_i} > 0$

The key idea of the binary search method is to iteratively estimate the global optimal value (p^*) of the fractional objective function of `P1`, instead of searching for it directly. Let $\mathscr{X}_f$ be the feasible region of `P1` (or `P2`). Given a real value r, we can know whether or not $r \leq p^*$ by checking

$$\exists \boldsymbol{x} \in \mathscr{X}_f, \text{ s.t. } rD(\boldsymbol{x}) - N(\boldsymbol{x}) \leq 0 \tag{7.2}$$

We now justify the correctness of the binary search method to solve any generic fractional programming problem $\max_{\boldsymbol{x}\in\mathcal{X}_f} N(\boldsymbol{x})/D(\boldsymbol{x})$ for any functions $N(\boldsymbol{x})$ and $D(\boldsymbol{x}) > 0$.

Lemma 1. *For any real value $r \in \mathscr{R}$, one of the following two conditions holds.*

(a) $r \leq p^* \Longleftrightarrow \exists \mathbf{x} \in \mathscr{X}_f$, *s.t.,* $rD(\mathbf{x}) - N(\mathbf{x}) \leq 0$

(b) $r > p^* \Longleftrightarrow \forall \mathbf{x} \in \mathscr{X}_f$, $rD(\mathbf{x}) - N(\mathbf{x}) > 0$

Proof. We only prove (a) as (b) is proven similarly.

'$\Leftarrow$': since $\exists x$ such that $rD(\boldsymbol{x}) \leq N(\boldsymbol{x})$, this means that $r \leq \frac{N(\boldsymbol{x})}{D(\boldsymbol{x})} \leq p^*$;

'$\Rightarrow$': Since `P1` optimizes a continuous objective over a closed convex set, then there exists an optimal solution $\boldsymbol{x}^*$ such that $p^* = \frac{N(\boldsymbol{x}^*)}{D(\boldsymbol{x}^*)} \geq r$ which rearranging gives the result. □ □

Algorithm 4: Binary Search

```
1  Input: ϵ, P_M and numRes;
2  (U_0, L_0) ← EstimateBounds(P_M, numRes);
3  (U, L) ← (U_0, L_0);
4  while U − L ≥ ϵ do
5  |  r ← (U+L)/2;
6  |  (feasible, x^r) ← CheckFeasibility(r);
7  |  if feasible then
8  |  |  L ← r
9  |  end
10 |  else
11 |  |  U ← r
12 |  end
13 end
14 return L, x^L;
```

Algorithm 5 describes the basic structure of the binary search method. Given the payoff matrix (P_M) and the total number of security resources ($numRes$), Algorithm 5 first initializes the upper bound (U_0) and lower bound (L_0) of the defender expected utility on Line 2. Then, in each iteration, r is set to be the mean of U and L. Line 6 checks whether the current r satisfies Equation (7.2). If so, $p^* \geq r$, the lower-bound of the binary search needs to be increased; in this case, it also returns a valid strategy x^r. Otherwise, $p^* < r$, the upper-bound of the binary search should be decreased. The search continues until the upper-bound and lower-bound are sufficiently close, i.e. $U - L < \epsilon$. The number of iterations in Algorithm 5 is bounded by $O(\log(\frac{U_0 - L_0}{\epsilon}))$. Specifically for SSGs we can estimate the upper and lower bounds as follows:

Lower bound: Let s_u be any feasible defender strategy. The defender utility based on using s_u against a adversary's quantal response is a lower bound of the optimal solution of `P1`. A simple example of s_u is the uniform strategy.

Upper bound: Since $P_i^d \leq U_i^d \leq R_i^d$ we have $U_i^d \leq \max_{i \in \mathcal{T}} R_i^d$. The defender's utility is computed as $\sum_{i \in \mathcal{T}} q_i U_i^d$, where U_i^d is the defender utility on target i and q_i is the probability that the adversary attacks target i. Thus, the maximum R_i^d serves as an upper bound of U_i^d.

We now turn to feasibility checking, which is performed in Step 6 in Algorithm 5. Given a real number $r \in \mathscr{R}$, in order to check whether Equation (7.2) is satisfied, we introduce `CF-OPT`.

$$\texttt{CF-OPT:} \quad \min_{\boldsymbol{x} \in \mathscr{X}_f} \; rD(\boldsymbol{x}) - N(\boldsymbol{x})$$

Let δ^* be the optimal objective function of the above optimization problem. If $\delta^* \leq 0$, Equation (7.2) must be true. Therefore, by solving the new optimization problem and checking if $\delta^* \leq 0$, we can answer if a given r is larger or smaller than the global maximum. However, the objective function in `CF-OPT` is still non-convex, therefore, solving it directly is still a hard problem. We introduce two methods to address this in the next two sections.

7.3 GOSAQ

We now present Global Optimal Strategy Against Quantal response (GOSAQ), which adapts Algorithm 5 to efficiently solve problems `P1` and `P2`. It does so through the following nonlinear invertible change of variables:

$$y_i = e^{-\beta_i x_i}, \forall i \in \mathcal{T} \tag{7.3}$$

7.3.1 GOSAQ with No Assignment Constraint

We first focus on applying GOSAQ to solve P1 for problems with no resource assignment constraints. Here, GOSAQ uses Algorithm 1, but with a *rewritten* CF−OPT as follows given the above variable substitution:

$$\min_{\mathbf{y}} \quad r\sum_{i\in\mathcal{T}}\theta_i y_i - \sum_{i\in\mathcal{T}}\theta_i P_i^d y_i + \sum_{i\in\mathcal{T}}\frac{\alpha_i\theta_i}{\beta_i} y_i \ln(y_i)$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{T}}\frac{-1}{\beta_i}\ln(y_i) \leq M \tag{7.4}$$

$$e^{-\beta_i} \leq y_i \leq 1, \qquad \forall i \tag{7.5}$$

Let's refer to the above optimization problem as GOSAQ−CP.

Lemma 2. *Let $Obj_{CF}(\mathbf{x})$ and $Obj_{GC}(\mathbf{y})$ be the objective function of CF−OPT and GOSAQ−CP respectively; $\mathscr{X}_f$ and $\mathscr{Y}_f$ denote the feasible domain of CF−OPT and GOSAQ−CP respectively:*

$$\min_{\mathbf{x}\in\mathscr{X}_f} Obj_{CF}(\mathbf{x}) = \min_{y\in\mathscr{Y}_f} Obj_{GC}(\mathbf{y}) \tag{7.6}$$

The proof, omitted for brevity, follows from the variable substitution in equation 7.6. Lemma 2 indicates that solving GOSAQ−CP is equivalent to solving CF−OPT. We now show that GOSAQ−CP is actually a convex optimization problem.

Lemma 3. *GOSAQ−CP is a convex optimization problem with a unique optimal solution.*

Proof. We can show that both the objective function and the nonlinear constraint function (7.4) in GOSAQ−CP are strictly convex by taking second derivatives and showing that the Hessian matrices are positive definite. The fact that the objective is strictly convex implies that it can have only one optimal solution. □ □

In theory, convex optimization problems like the one above, can be solved in polynomial time through the ellipsoid method or interior point method with the volumetric barrier function [Boyd and Vandenberghe, 2004] (in practice there are a number of nonlinear solvers capable of finding the only KKT point efficiently). Hence, GOSAQ entails running Algorithm 5, performing Step 6 with $O(\log(\frac{U_0-L_0}{\epsilon}))$ times, and each time solving `GOSAQ-CP` which is polynomial solvable. Therefore, GOSAQ is a polynomial time algorithm.

We now show the bound of GOSAQ's solution quality.

Lemma 4. *Let L^* and U^* be the lower and upper bounds of* GOSAQ *when the algorithm stops, and* $\mathbf{x}^*$ *is the defender strategy returned by* GOSAQ*. Then,*

$$L^* \leq Obj_{P1}(\mathbf{x}^*) \leq U^*$$

where $Obj_{P1}(\mathbf{x})$ denotes the objective function of `P1`*.*

Proof. Given r, Let $\delta^*(r)$ be the minimum value of the objective function in `GOSAQ-CP`. When GOSAQ stops, we have $\delta^*(L^*) \leq 0$, because from Lines 6-8 of Algorithm 5, updating the lower bound requires it. Hence, from Lemma 2, $L^*D(\boldsymbol{x}^*) - N(\boldsymbol{x}^*) \leq 0 \Rightarrow L^* \leq \frac{N(\boldsymbol{x}^*)}{D(\boldsymbol{x}^*)}$. Similarly, $\delta^*(U^*) \geq 0 \Rightarrow U^* > \frac{N(\boldsymbol{x}^*)}{D(\boldsymbol{x}^*)}$ □ □

Theorem 1. *Let* $\mathbf{x}^*$ *be the defender strategy computed by* GOSAQ*,*

$$0 \leq p^* - Obj_{P1}(\mathbf{x}^*) \leq \epsilon \tag{7.7}$$

Proof. p^* is the global maximum of `P1`, so $p^* \geq Obj_{P1}(\boldsymbol{x}^*)$. Let L^* and U^* be the lower and upper bound when GOSAQ stops. Based on Lemma 4, $L^* \leq Obj_{P1}(\boldsymbol{x}^*) \leq U^*$. Simultaneously, Algorithm 5 indicates that $L^* \leq p^* \leq U^*$.

Therefore, $0 \leq p^* - Obj_{P1}(\boldsymbol{x}^*) \leq U^* - L^* \leq \epsilon$ □ □

Theorem 1 indicates that the solution obtained by GOSAQ is an ϵ-optimal solution.

7.3.2 GOSAQ with Assignment Constraints

In order to address the assignment constraints, we need to solve `P2`. Note that the objective function of `P2` is the same as that of `P1`. The difference lies in the extra constraints which enforce the marginal coverage to be feasible. Therefore we once again use Algorithm 5 with variable substitution given in Equation 7.3, but modify `GOSAQ-CP` as follows (which is referred as `GOSAQ-CP-C`) to incorporate the extra constraints:

$$\min_{\boldsymbol{y},a} \quad r\sum_{i\in\mathcal{T}}\theta_i y_i - \sum_{i\in\mathcal{T}}\theta_i P_i^d y_i + \sum_{i\in\mathcal{T}}\frac{\alpha_i\theta_i}{\beta_i}y_i\ln(y_i)$$

$$\text{s.t.} \quad \texttt{Constraint } (7.4), (7.5)$$

$$\frac{-1}{\beta_i}\ln(y_i) = \sum_{A_j\in\mathcal{A}} a_j A_{ij}, \quad \forall i \in \mathcal{T} \tag{7.8}$$

$$\sum_{A_j\in\mathcal{A}} a_j = 1 \tag{7.9}$$

$$0 \le a_j \le 1, \quad A_j \in \mathcal{A} \tag{7.10}$$

Equation (7.8) is a nonlinear equality constraint that makes this optimization problem non-convex. There are no known polynomial time algorithms for generic non-convex optimization problems, which can have multiple local minima. We can attempt to solve such non-convex problems using one of the efficient nonlinear solvers but we would obtain a KKT point which can be only locally optimal. There are a few research grade global solvers for non-convex programs, however they are limited to solving specific problems or small instances. Therefore, in the presence of assignment constraints, GOSAQ is no longer guaranteed to return the optimal solution as we might be left with locally optimal solutions when solving the subproblems `GOSAQ-CP-C`.

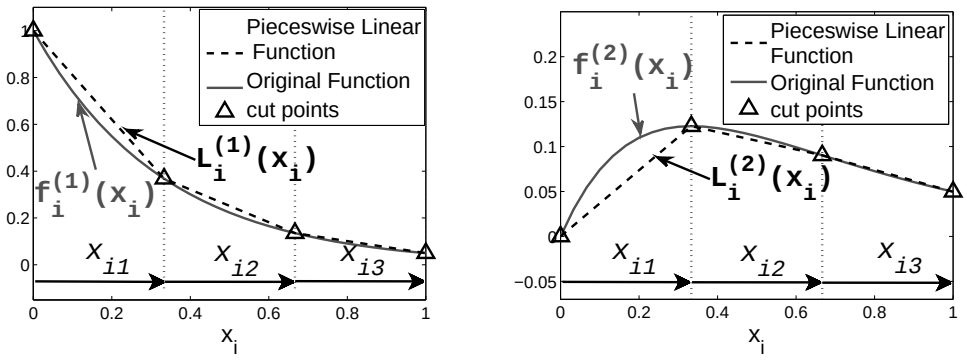
7.4 PASAQ

Since GOSAQ may be unable to provide a quality bound in the presence of assignment constraints (and as shown later, may turn out to be inefficient in such cases), we propose the Piecewise linear Approximation of optimal Strategy Against Quantal response (PASAQ). PASAQ is an algorithm to compute the approximate optimal defender strategy. PASAQ has the same structure as Algorithm 5. The key idea in PASAQ is to use a piecewise linear function to approximate the nonlinear objective function in `CF-OPT`, and thus convert it into a Mixed-Integer Linear Programming (MILP) problem. Such a problem can easily include assignment constraints giving an approximate solution for a SSG against a QR-adversary with assignment constraints.

In order to demonstrate the piecewise approximation in PASAQ, we first rewrite the nonlinear objective function of `CF-OPT` as:

$$\sum_{i \in \mathcal{T}} \theta_i (r - P_i^d) e^{-\beta_i x_i} - \sum_{i \in \mathcal{T}} \theta_i \alpha_i x_i e^{-\beta_i x_i}$$

The goal is to approximate the two nonlinear function $f_i^{(1)}(x_i) = e^{-\beta_i x_i}$ and $f_i^{(2)}(x_i) = x_i e^{-\beta_i x_i}$ as two piecewise linear functions in the range $x_i \in [0, 1]$, for each $i = 1..|\mathcal{T}|$. We first uniformly



(a) approximation of $f_i^{(1)}(x_i)$ (b) approximation of $f_i^{(2)}(x_i)$

Figure 7.1: Piecewise Linear Approximation

divide the range $[0, 1]$ into K pieces (segments). Simultaneously, we introduce a set of new variables $\{x_{ik}, k = 1..K\}$ to represent the portion of x_i in each of the K pieces, $\{[\frac{k-1}{K}, \frac{k}{K}], k = 1..K\}$. Therefore, $x_{ik} \in [0, \frac{1}{K}], \forall k = 1..K$ and $x_i = \sum_{k=1}^{K} x_{ik}$. In order to ensure that $\{x_{ik}\}$ is a valid partition of x_i, all x_{ik} must satisfy: $x_{ik} > 0$ only if $x_{ik'} = \frac{1}{K}, \forall k' < k$. In other words, x_{ik} can be non-zero only when all the previous pieces are completely filled. Figures 7.1(a) and 7.1(b) display two examples of such a partition.

Thus, we can represent the two nonlinear functions as piecewise linear functions using $\{x_{ik}\}$. Let $\{(\frac{k}{K}, f_i^{(1)}(\frac{k}{K})), k = 0..K\}$ be the $K+1$ cut-points of the linear segments of function $f_i^{(1)}(x_i)$, and $\{\gamma_{ik}, k = 1..K\}$ be the slopes of each of the linear segments. Starting from $f_i^{(1)}(0)$, the piecewise linear approximation of $f_i^{(1)}(x_i)$, denoted as $L_i^{(1)}(x_i)$:

$$L_i^{(1)}(x_i) = f_i^{(1)}(0) + \sum_{k=1}^{K} \gamma_{ik} x_{ik} = 1 + \sum_{k=1}^{K} \gamma_{ik} x_{ik}$$

Similarly, we can obtain the piecewise linear approximation of $f_i^{(2)}(x_i)$, denoted as $L_i^{(2)}(x_i)$:

$$L_i^{(2)}(x_i) = f_i^{(2)}(0) + \sum_{k=1}^{K} \mu_{ik} x_{ik} = \sum_{k=1}^{K} \mu_{ik} x_{ik}$$

where, $\{\mu_{ik}, k = 1..K\}$ is the slope of each linear segment.

7.4.1 PASAQ with No Assignment Constraint

In domains without assignment constraints, PASAQ consists of Algorithm 5, but with CF−OPT rewritten as follows:

$$\min_{\boldsymbol{x},z} \sum_{i\in\mathcal{T}} \theta_i(r - P_i^d)(1 + \sum_{k=1}^{K} \gamma_{ik} x_{ik}) - \sum_{i\in\mathcal{T}} \theta_i \alpha_i \sum_{k=1}^{K} \mu_{ik} x_{ik}$$

$$\text{s.t.} \sum_{i\in\mathcal{T}} \sum_{k=1}^{K} x_{ik} \leq M \tag{7.11}$$

$$0 \leq x_{ik} \leq \frac{1}{K}, \quad \forall i, \quad k = 1 \dots K \tag{7.12}$$

$$z_{ik} \frac{1}{K} \leq x_{ik}, \quad \forall i, \quad k = 1 \dots K-1 \tag{7.13}$$

$$x_{i(k+1)} \leq z_{ik}, \quad \forall i, \quad k = 1 \dots K-1 \tag{7.14}$$

$$z_{ik} \in \{0, 1\}, \quad \forall i, \quad k = 1 \dots K-1 \tag{7.15}$$

Let's refer to the above MILP formulation as PASAQ−MILP.

Lemma 5. *The feasible region for* $\mathbf{x} = \langle x_i = \sum_{k=1}^{K} x_{ik}, i \in \mathcal{T} \rangle$ *of PASAQ−MILP is equivalent to that of P1*

JUSTIFICATION. The auxiliary integer variable z_{ik} indicates whether or not $x_{ik} = \frac{1}{K}$. Equation (7.13) enforces that $z_{ik} = 0$ only when $x_{ik} < \frac{1}{K}$. Simultaneously, Equation (7.14) enforces that $x_{i(k+1)}$ is positive only if $z_{ik} = 1$. Hence,$\{x_{ik}, k = 1..K\}$ is a valid partition of x_i and $x_i = \sum_{k=1}^{K} x_{ik}$ and that $x_i \in [0, 1]$. Thus, the feasible region of PASAQ−MILP is equivalent to P1 □

Lemma 5 shows that the solution provided by PASAQ is in the feasible region of P1. However, PASAQ approximates the minimum value of CF−OPT by using PASAQ−MILP, and furthermore solves P1 approximately using binary search. Hence, we need to show an error bound on the solution quality of PASAQ.

Table 7.2: Notations for Error Bound Proof

$\underline{\theta} := \min_{i \in \mathcal{T}} \theta_i$	$\overline{R^d} := \max_{i \in \mathcal{T}} \lvert R_i^d \rvert$	$\overline{\beta} := \max_{i \in \mathcal{T}} \beta_i$
$\overline{\theta} := \max_{i \in \mathcal{T}} \theta_i$	$\overline{P^d} := \max_{i \in \mathcal{T}} \lvert P_i^d \rvert$	$\overline{\alpha} := \max_{i \in \mathcal{T}} \alpha_i$

Table 7.3: Game Constant

$\overline{\theta} := \max_{i \in \mathcal{T}} \theta_i$	$\underline{\theta} := \min_{i \in \mathcal{T}} \theta_i$
$\overline{R^d} := \max_{i \in \mathcal{T}} \lvert R_i^d \rvert$	$\overline{P^d} := \max_{i \in \mathcal{T}} \lvert P_i^d \rvert$
$\overline{\beta} := \max_{i \in \mathcal{T}} \beta_i$	$\overline{\alpha} := \max_{i \in \mathcal{T}} \alpha_i$

We first show Lemma 7-9 on the way to build the proof for the error bound. Full proofs are available in the Appendix A. We also define the game constants decided by the payoff in Table A.1. Further, we define two constants which are decided by the game payoffs: $C_1 = (\overline{\theta}/\underline{\theta})e^{\overline{\beta}}\{(\overline{R^d} + \overline{P^d})\overline{\beta} + \overline{\alpha}\}$ and $C_2 = 1 + (\overline{\theta}/\underline{\theta})e^{\overline{\beta}}$. The notation used is defined in Table A.1. In the following, we are interested in obtaining a bound on the difference between p^* (the global optimal obtained from `P1`) and $Obj_{P1}(\tilde{\boldsymbol{x}}^*)$, where $\tilde{\boldsymbol{x}}^*$ is the strategy obtained from PASAQ. However, along the way, we have to obtain a bound for the difference between $Obj_{P1}(\tilde{\boldsymbol{x}}^*)$ and its corresponding piecewise linear approximation $\tilde{Obj}_{P1}(\tilde{\boldsymbol{x}}^*)$.

Lemma 6. *Let $\tilde{L}^*$ and $\tilde{U}^*$ be the final lower and upper bounds of* PASAQ*, and $\tilde{\mathbf{x}}^*$ is the defender strategy returned by* PASAQ*. Then,*

$$\tilde{L}^* \leq \tilde{Obj}_{P1}(\tilde{\mathbf{x}}^*) \leq \tilde{U}^*$$

Lemma 7. *Let $\tilde{N}(\mathbf{x}) = \sum_{i \in \mathcal{T}} \theta_i \alpha_i L_i^{(2)}(x_i) + \sum_{i \in \mathcal{T}} \theta_i P_i^d L_i^{(1)}(x_i)$ and $\tilde{D}(\mathbf{x}) = \sum_{i \in \mathcal{T}} \theta_i L_i^{(1)}(x_i) > 0$ be the piecewise linear approximation of $N(\mathbf{x})$ and $D(\mathbf{x})$ respectively. Then, $\forall \mathbf{x} \in \mathscr{X}_f$*

$$|N(\mathbf{x}) - \tilde{N}(\mathbf{x})| \leq (\overline{\theta}\overline{\alpha} + \overline{P^d}\overline{\theta}\overline{\beta})\frac{|\mathcal{T}|}{K}$$

$$|D(\mathbf{x}) - \tilde{D}(\mathbf{x})| \leq \overline{\theta}\overline{\beta}\frac{|\mathcal{T}|}{K}$$

Lemma 8. *The difference between the objective funciton of* P1*,* $Obj_{P1}(\mathbf{x})$*, and its corresponding piecewise linear approximation,* $\tilde{Obj}_{P1}(\mathbf{x})$*, is less than* $C_1\frac{1}{K}$

Proof.

$$
\begin{aligned}
|Obj_{P1}(\boldsymbol{x}) - \tilde{Obj}_{P1}(\boldsymbol{x})| &= |\frac{N(\boldsymbol{x})}{D(\boldsymbol{x})} - \frac{\tilde{N}(\boldsymbol{x})}{\tilde{D}(\boldsymbol{x})}| \\
&= |\frac{N(\boldsymbol{x})}{D(\boldsymbol{x})} - \frac{N(\boldsymbol{x})}{\tilde{D}(\boldsymbol{x})} + \frac{N(\boldsymbol{x})}{\tilde{D}(\boldsymbol{x})} - \frac{\tilde{N}(\boldsymbol{x})}{\tilde{D}(\boldsymbol{x})}| \\
&\leq \frac{1}{\tilde{D}(\boldsymbol{x})}(|Obj_{P1}(\boldsymbol{x})||D(\boldsymbol{x}) - \tilde{D}(\boldsymbol{x})| + |N(\boldsymbol{x}) - \tilde{N}(\boldsymbol{x})|)
\end{aligned}
$$

Based on Lemma 7, $|Obj_{P1}(\boldsymbol{x})| \leq \overline{R^d}$, and $\tilde{D}(\boldsymbol{x}) \geq |\mathcal{T}|\underline{\theta}e^{-\overline{\beta}}$.

$$
|Obj_{P1}(\boldsymbol{x}) - \tilde{Obj}_{P1}(\boldsymbol{x})| \leq C_1\frac{1}{K}
$$

□

Lemma 9. *Let* $\tilde{L}^*$ *and* L^* *be final lower bound of* PASAQ *and* GOSAQ*,*

$$
L^* - \tilde{L}^* \leq C_1\frac{1}{K} + C_2\epsilon
$$

Theorem 2. *Let* $\tilde{x}^*$ *be the defender strategy computed by* PASAQ*,* p^* *is the global optimal defender expected utility,*

$$
0 \leq p^* - Obj_{P1}(\tilde{x}^*) \leq 2C_1\frac{1}{K} + (C_2 + 1)\epsilon
$$

Proof. The first inequality is implied since $\tilde{x}^*$ is a feasible solution. Furthermore,

$$
\begin{aligned}
p^* - Obj_{P1}(\tilde{x}^*) =&(p^* - L^*) + (L^* - \tilde{L}^*) + (\tilde{L}^* - \tilde{Obj}_{P1}(\tilde{x}^*)) \\
&+ (\tilde{Obj}_{P1}(\tilde{x}^*) - Obj_{P1}(\tilde{x}^*))
\end{aligned}
$$

Algorithm 5 indicates that $L^* \leq p^* \leq U^*$, hence $p^* - L^* \leq \epsilon$. Additionally, Lemma 8, 21 and 6 provide an upper bound on $\tilde{Obj}_{P1}(\tilde{x}^*) - Obj_{P1}(\tilde{x}^*)$, $L^* - \tilde{L}^*$ and $\tilde{L}^* - \tilde{Obj}_{P1}(\tilde{x}^*)$, therefore

$$p^* - Obj_{P1}(\tilde{x}^*) \leq \epsilon + C_1 \frac{1}{K} + C_2\epsilon + C_1 \frac{1}{K} \leq 2C_1 \frac{1}{K} + (C_2 + 1)\epsilon$$

□

Theorem 2 suggests that, given a game instance, the solution quality of PASAQ is bounded linearly by the binary search threshold ϵ and the piecewise linear accuracy $\frac{1}{K}$. Therefore the PASAQ solution can be made arbitrarily close to the optimal solution with sufficiently small ϵ and sufficiently large K.

7.4.2 PASAQ With Assignment Constraints

In order to extend PASAQ to handle the assignment constraints, we need to modify `PASAQ-MILP` as the follows, referred to as `PASAQ-MILP-C`,

$$\min_{\boldsymbol{x},z,a} \sum_{i\in\mathcal{T}} \theta_i (r - P_i^d)(1 + \sum_{k=1}^{K} \gamma_{ik} x_{ik}) - \sum_{i\in\mathcal{T}} \theta_i \alpha_i \sum_{k=1}^{K} \mu_{ik} x_{ik}$$

s.t. `Constraint` $(7.11) - (7.15)$

$$\sum_{k=1}^{K} x_{ik} = \sum_{A_j \in \mathcal{A}} a_j A_{ij}, \quad \forall i \in \mathcal{T} \tag{7.16}$$

$$\sum_{A_j \in \mathcal{A}} a_j = 1 \tag{7.17}$$

$$0 \leq a_j \leq 1, \quad A_j \in \mathcal{A} \tag{7.18}$$

`PASAQ-MILP-C` is an MILP so it can be solved optimally with any MILP solver (e.g. CPLEX). We can prove, similarly as we did for Lemma 5, that the above MILP formulation has the same feasible region as `P2`. Hence, it leads to a feasible solution of `P2`. Furthermore, the error bound

of PASAQ relies on the approximation accuracy of the objective function by the piecewise linear function and the fact that the subproblem `PASAQ-MILP-C` can be solved optimally. Both conditions have not changed from the cases without assignment constraints to the cases with assignment constraints. Hence, the error bound is the same as that shown in Theorem 2.

7.5 Experiments

We separate our experiments into two sets: the first set focuses on the cases where there is no constraint on assigning the resources; the second set focuses on cases with assignment constraints. In both sets, we compare the solution quality and runtime of the two new algorithms, GOSAQ and PASAQ, with the previous benchmark algorithm BRQR. The results were obtained using CPLEX to solve the MILP for PASAQ. For both BRQR and GOSAQ, we use the MATLAB toolbox function `fmincon` to solve nonlinear optimization problems[1]. All experiments were conducted on a standard 2.00GHz machine with 4GB main memory. For each setting of the experiment parameters (i.e. number of targets, amount of resources and number of assignment constraints), we tried 50 different game instances. In each game instance, payoffs R_i^d and R_i^a are chosen uniformly randomly from 1 to 10, while P_i^d and P_i^a are chosen uniformly randomly from -10 to -1; feasible assignments A_j are generated by randomly setting each element A_{ij} to 0 or 1. For the parameter λ of the quantal response in Equation (7.1), we used the same value ($\lambda = 0.76$) as learned in the experiment in Chapter 4.

[1] We also tried the KNITRO [Byrd et al., 2006] solver. While it gave the same solution quality as `fmincon`, it was three-times slower than `fmincon`; as a result we report results with `fmincon`

7.5.1 No Assignment Constraints

We first present experimental results comparing the solution quality and runtime of the three algorithms (GOSAQ,PASAQ and BRQR) in cases without assignment constraints.

Solution Quality: For each game instance, GOSAQ provides the ϵ-optimal defender expected utility, BRQR presents the best local optimal solution among all the local optimum it finds, and PASAQ leads to an approximated global optimal solution. We measure the solution quality of different algorithms using average defender's expected utility over all the 50 game instances.

Figures 7.2(a), 7.2(c) and 7.2(e) show the solution quality results of different algorithms under different conditions. In all three figures, the average defender expected utility is displayed on the y-axis. On the x-axis, Figure 7.2(a) changes the numbers of targets ($|\mathcal{T}|$) keeping the ratio of resources (M) to targets and ϵ fixed as shown in the caption; Figure 7.2(c) changes the ratio of resources to targets fixing targets and ϵ as shown; and Figure 7.2(e) changes the value of the binary search threshold ϵ. Given a setting of the parameters ($|\mathcal{T}|$, M and ϵ), the solution qualities of different algorithms are displayed in a group of bars. For example, in Figure 7.2(a), $|\mathcal{T}|$ is set to 50 for the leftmost group of bars, M is 5 and $\epsilon = 0.01$. From left to right, the bars show the solution quality of BRQR (with 20 and 100 iterations), PASAQ (with 5,10 and 20 pieces) and GOSAQ.

Key observations from Figures 7.2(a), 7.2(c) and 7.2(e) include: (i) The solution quality of BRQR drops quickly as the number of targets increases; increasing the number of iterations in BRQR improves the solution quality, but the improvement is very small. (ii) The solution quality of PASAQ improves as the number of pieces increases; and it converges to the GOSAQ solution as the number of pieces becomes larger than 10. (iii) As the number of resources increases, the

defender expected utility also increases; and the resource count does not impact the relationship of solution quality between different algorithms. (iv) As ϵ becomes smaller, the solution quality of both GOSAQ and PASAQ improves. However, after epsilon becomes sufficiently small (≤ 0.1), no substantial improvement is achieved by further decreasing the value of ϵ. In other words, the solution quality of both GOSAQ and PASAQ converges.

In general, BRQR has the worst solution quality; GOSAQ has the best solution quality. PASAQ achieves almost the same solution quality as GOSAQ when it uses more than 10 pieces.

Runtime: We present the runtime results in Figures 7.2(b), 7.2(d) and 7.2(f). In all three figures, the y-axis display the runtime, the x-axis displays the variables which we vary to measure their impact on the runtime of the algorithms. For BRQR run time is the sum of the run-time across all its iterations.

Figure 7.2(b) shows the change in runtime as the number of targets increases. The number of resources and the value of ϵ are shown in the caption. BRQR with 100 iterations is seen to run significantly slower than GOSAQ and PASAQ. Figure 7.2(d) shows the impact of the ratio of resource to targets on the runtime. The figure indicates that the runtime of the three algorithms is independent of the change in the number of resources. Figure 7.2(f) shows how runtime of GOSAQ and PASAQ is affected by the value of ϵ. On the x-axis, the value for ϵ decreases from left to right. The runtime increases linearly as ϵ decreases exponentially. In both Figures 7.2(d) and 7.2(f), the number of targets and resources are displayed in the caption.

Overall, the results suggest that GOSAQ is the algorithm of choice when the domain has no assignment constraints. Clearly, BRQR has the worst solution quality, and it is the slowest of the set of algorithms. PASAQ has a solution quality that approaches that of GOSAQ when the number of pieces is sufficiently large (≥ 10), and GOSAQ and PASAQ also achieve comparable

runtime efficiency. Thus, in cases with no assignment constraints, PASAQ offers no advantages over GOSAQ.

7.5.2 With Assignment Constraints

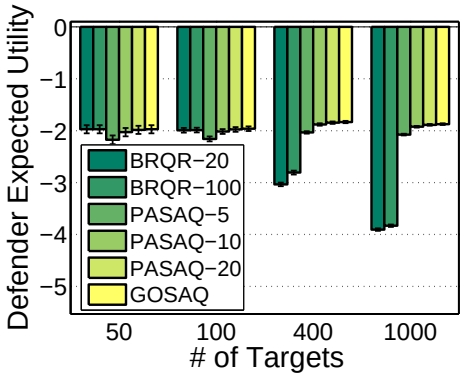
In the second set, we introduce assignment constraints into the problem. The feasible assignments are randomly generated. We present experimental results on both solution quality and runtime.

Solution Quality: Figures 7.3(a) and 7.3(b) display the solution quality of the three algorithms with varying number of targets ($|\mathcal{T}|$) and varying number of feasible assignments ($|\mathcal{A}|$). In both figures, the average defender expected utility is displayed on the y-axis. In Figure 7.3(a) the number of targets is displayed on the x-axis, and the ratio of $|\mathcal{A}|$ to $|\mathcal{T}|$ is set to 60. BRQR is seen to have very poor performance. Furthermore, there is very little gain in solution quality from increasing its number of iterations. While GOSAQ provides the best solution quality, PASAQ achieves almost identical solution quality when the number of pieces is sufficiently large (> 10). Figure 7.3(b) shows how solution quality is impacted by the number of feasible assignments, which is displayed on the x-axis. Specifically, the x-axis shows numbers of assignment constraints $\mathcal{A}$ to be 20 times, 60 times and 100 times the number of targets. The number of targets is set to 60. Once again, BRQR has significantly lower solution quality, and it drops as the number of assignments increases; and PASAQ again achieves almost the same solution quality as GOSAQ, as the number the number of pieces is larger than 10.

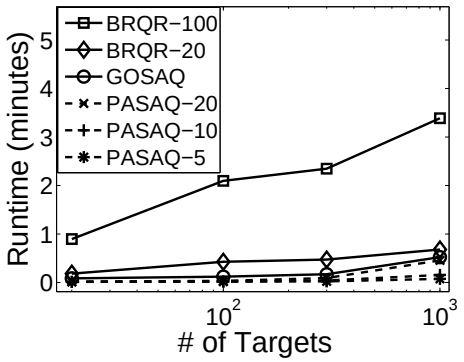
Runtime: We present the runtime results in Figures 7.3(c), 7.3(e), 7.3(d) and 7.3(f). In all experiments, we set 80 minutes as the cut-off. Figure 7.3(c) displays the runtime on the y-axis and the number of targets on the x-axis. It is clear that GOSAQ runs significantly slower than both PASAQ and BRQR, and slows down exponentially as the number of targets increases. Figure

7.3(e) shows extended runtime result of BRQR and PASAQ as the number of targets increases. PASAQ runs in less than 4 minutes with 200 targets and 12000 feasible assignments. BRQR runs significantly slower with higher number of iterations.

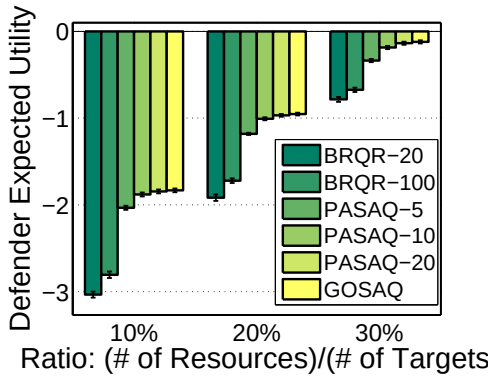
Overall, the results suggest that PASAQ is the algorithm of choice when the domain has assignment constraints. Clearly, BRQR has significantly lower solution quality than PASAQ. PASAQ not only has a solution quality that approaches that of GOSAQ when the number of pieces is sufficiently large ($\geq$ 10), PASAQ is significantly faster than GOSAQ (which suffers exponential slowdown with scale-up in the domain).



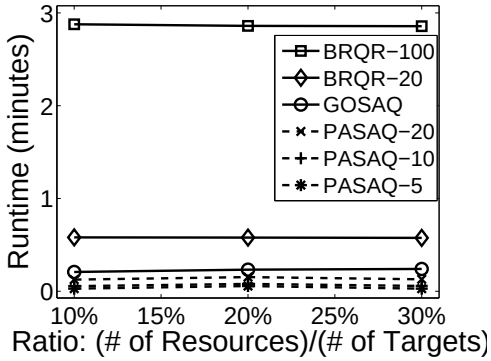
(a) Solution Quality v.s. $|\mathcal{T}|$ ($M = 0.1|\mathcal{T}|$, $\epsilon = 0.01$)



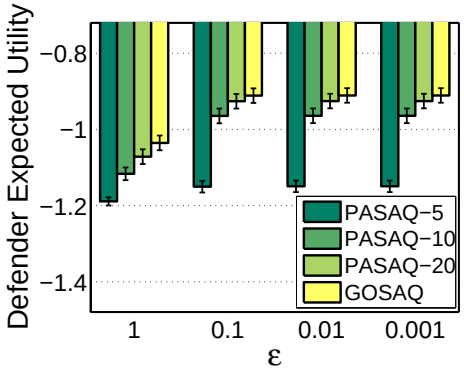
(b) Runtime v.s. $|\mathcal{T}|$ ($M = 0.1|\mathcal{T}|$, $\epsilon = 0.01$)



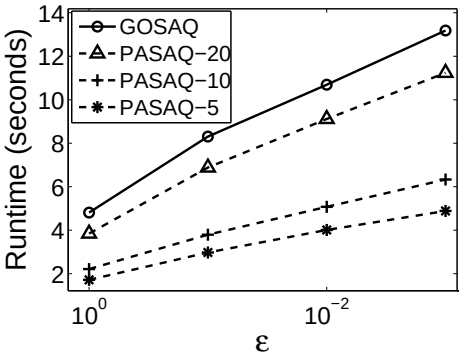
(c) Solution Quality v.s. M ($|\mathcal{T}| = 400$, $\epsilon = 0.01$)



(d) Runtime v.s. M($|\mathcal{T}| = 400$, $\epsilon = 0.01$)

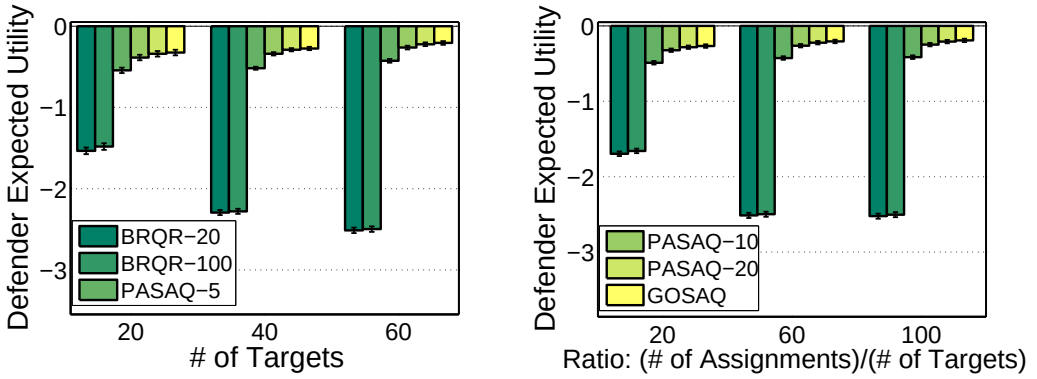


(e) Solution Quality v.s. ϵ ($|\mathcal{T}| = 400$, $M = 80$)



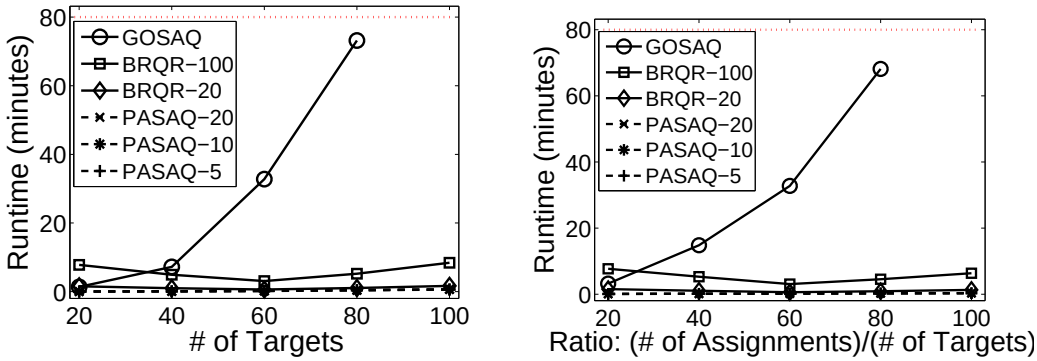
(f) Runtime v.s. ϵ ($|\mathcal{T}| = 400$, $M = 80$)

Figure 7.2: Solution Quality and Runtime Comparison, without assignment constraints (better in color)



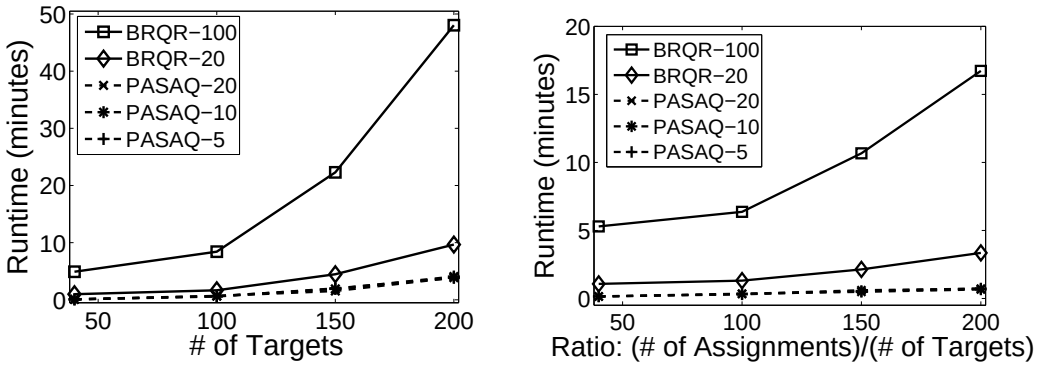
(a) Solution Quality v.s. $|\mathcal{T}|$ ($|\mathcal{A}| = 60|\mathcal{T}|$)

(b) Solution Quality v.s. $|\mathcal{A}|$ ($|\mathcal{T}| = 60$)



(c) Runtime v.s. $|\mathcal{T}|$ ($|\mathcal{A}| = 60|\mathcal{T}|$)

(d) Runtime v.s. $|\mathcal{A}|$ ($|\mathcal{T}| = 60$)



(e) Runtime v.s. $|\mathcal{T}|$ ($|\mathcal{A}| = 60|\mathcal{T}|$)

(f) Runtime v.s. $|\mathcal{A}|$ ($|\mathcal{T}| = 60$)

Figure 7.3: Solution Quality and Runtime Comparison, with assignment constraint (better in color)

Chapter 8: Scaling-up

This chapter focuses on scaling up SSG algorithms integrated with any of a family of discrete choice models[Train, 2003; Goeree et al., 2005], an important class of bounded rationality models of adversary decision making, of which quantal response model is an important representative. Unfortunately, PASAQ fails to scale-up when faced with massive scale since it requires explicit enumeration of defender strategies, which is not feasible in massive-scale SSGs such as with the FAMS or the US Coast Guard in bigger ports. Previous work has provided branch-and-price (BnP) [Barnhart et al., 1994] as a key technique to avoid explicit enumeration of defender strategies in SSGs[Jain et al., 2010a]; however, how well BnP would handle bounded rationality models is an unknown.

In this chapter, I present a novel algorithm called BLADE to scale-up SSGs with complex adversary models. In Section 8.1, I extend the PASAQ algorithm to handle any of the family of the discrete choice model. Section 8.2 investigates the effectiveness of BnP in SSG algorithms handling bounded rationality via a BnP algorithm called COCOMO. As I will show in more details, the non-convexity of the objective function given bounded rationality adversary models creates enormous hurdles in scale-up. In Section 8.3, I provide a new algorithm called BLADE, which for the first time illustrates an efficient realization of the cutting-plane approach in SSGs[Kelley,

1960; Boyd and Vandenberghe, 2008]. The cutting-planes approach iteratively refines the solution space via cuts. Our key hypothesis is that with these cuts, BLADE can successfully exploit the structure of the solution space of defender strategies – generated due to the bounded rationality adversary models in SSGs – whereas BnP approaches are blind to this structure. BLADE is based on three novel ideas. First, i present a separation oracle that can effectively prune the search space via deep cuts. More importantly I show that to handle massive scale SSGs, not only must this separation oracle itself use a secondary oracle but that this two-level hierarchy of oracles is efficient. Second, I provide a novel heuristic to further speed-up BLADE by exploiting the SSG objective function to improve its cuts. Third, BLADE provides a technique for quality-efficiency tradeoff. Finally, in section 8.4, I present the experimental results on comparing COCOMO and BLADE and show that BLADE is significantly more efficient than COCOMO.

8.1 Generalized PASAQ

Before discussing scale-up, we generalize the PASAQ algorithm to solve SSGs integrated with bounded rationality models, as they are specialized to the QR model. In PASAQ, the objective function of `P1` is

$$F(\boldsymbol{x}) = \sum_i q_i(\boldsymbol{x}) U_i^d(x_i) = \sum_i \frac{e^{\lambda U_i^a(x_i)}}{\sum_j e^{\lambda U_j^a(x_j)}} U_i^d(x_i)$$

QR is a representative of a more general form of the discrete choice model [Train, 2003; Goeree et al., 2005] for adversary response as shown in Equation (8.1). In SSGs, typically $f_i(x_i) \geq 0, \forall x_i \in [0,1]$ is a monotonically decreasing function of x_i, indicating that as the

defender's marginal coverage on target i increases, the probability that the adversary chooses this target decreases, e.g., in QR, $f_i(x_i) = e^{\lambda U_i^a(x_i)}$ is an exponentially decreasing function of x_i.

$$q_i(\boldsymbol{x}) = \frac{f_i(x_i)}{\sum_i f_i(x_i)} \tag{8.1}$$

Furthermore, PASAQ handles the constraints on the defender resource allocation by enumerating all the possible assignments. In general, there are *spatio-temporal constraints*: the air marshal's two flights have to be connected, e.g., an air marshal cannot fly from Los Angeles to New York and then from Chicago to Seattle. Moreover, the second flight cannot depart before the first flight arrives. Furthermore, there might be *user-specified constraints* [An et al., 2010]: FAMS might want to cover 50% of the flights to Chicago; or that at most 30% of the flights departing from the JFK airport to be covered. Thus, the defender's optimization problem can be written as follows:

$$\texttt{P1}: \{\max_{\boldsymbol{x}} \sum_{i \in \mathcal{T}} U_i^d(x_i) q_i(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{X}_f \equiv \mathcal{X}_{f_1} \cap \mathcal{X}_{f_2}\}$$

$$\mathcal{X}_{f_1} := \{\boldsymbol{x} \mid \boldsymbol{x} = \sum_{A_j \in \mathcal{A}} a_j A_j, \sum_j a_j = 1, a_j \in [0,1]\} \tag{8.2}$$

$$\mathcal{X}_{f_2} := \{\boldsymbol{x} \mid B\boldsymbol{x} \leq \boldsymbol{b}\} \tag{8.3}$$

We denote the objective function in P1 as $F(\boldsymbol{x})$. In $F(\boldsymbol{x})$, $U_i^d(x_i)$ is the defender's expected utility if the adversary chooses target i; and $q_i(\boldsymbol{x})$ is the probability that the adversary chooses target i. $q_i(\boldsymbol{x})$ depends on the model used, e.g., assuming a QR model of the adversary, $q_i(\boldsymbol{x})$ is a logit function of $\boldsymbol{x}$. This leads to a nonlinear fractional optimization problem, which in general is NP-hard [Vavasis, 1995]. Furthermore, $\mathcal{X}_f$ represents the feasible region of the marginal coverage vector, which is defined by the intersection of $\mathcal{X}_{f_1}$ which emcompasses the spatio-temporal

constraints, and $\mathcal{X}_{f_2}$ which emcompasses the user-specified linear constraints on the marginals. Therefore, $\mathcal{X}_f = \mathcal{X}_{f1} \cap \mathcal{X}_{f2}$.

G-PASAQ generalizes PASAQ to solve P1 with the general form of $q_i(\boldsymbol{x})$ in Equation (8.1). As with PASAQ, G-PASAQ solves this non-linear fractional optimization problem using binary search. At each step of the binary search it solves a non-convex optimization problem whose objective is a sum of nonlinear functions of marginal variables x_i. Approximating each of these single-variable nonlinear functions as a piecewise-linear function with K segments, the non-convex problem is approximated by the MILP shown in Equation (8.4) - (8.9); this MILP solved in each iteration of the binary search.

$$\min_{\boldsymbol{x},z} \sum_{i\in\mathcal{T}}(r - P_i^d)(f_i(0) + \sum_{k=1}^{K}\gamma_{ik}x_{ik}) - \sum_{i\in\mathcal{T}}\alpha_i\sum_{k=1}^{K}\mu_{ik}x_{ik} \tag{8.4}$$

$$\text{s.t. } 0 \le x_{ik} \le 1/K, \quad \forall i, \quad k = 1 \dots K \tag{8.5}$$

$$z_{ik}/K \le x_{ik}, \quad \forall i, \quad k = 1 \dots K-1 \tag{8.6}$$

$$x_{i(k+1)} \le z_{ik}, \quad \forall i, \quad k = 1 \dots K-1 \tag{8.7}$$

$$z_{ik} \in \{0,1\}, \quad \forall i, \quad k = 1 \dots K-1 \tag{8.8}$$

$$\boldsymbol{x} \in \mathcal{X}_f \tag{8.9}$$

The objective function in Equation (8.4) is a piecewise linear approximation of $\sum_{i\in\mathcal{T}}(r - P_i^d)f_i(x_i) - \sum_{i\in\mathcal{T}}\alpha_i x_i f_i(x_i)$ where $\alpha_i = R_i^d - P_i^d$ is a constant, γ_{ik} is the slope of $f_i(x_i)$ in the k^{th} segments and μ_{ik} is the corresponding slope of $x_i f_i(x_i)$. The range of each x_i is divided into K segments, and x_i is replaced by the variables $\{x_{ik}, k = 1 \dots k\}$ such that $x_i = \sum_{k=1}^{K} x_{ik}$. $\{z_{ik}, k = 1..K\}$ in Equation (8.6)-(8.8) are integer variables that decide the particular segment that x_i lies in. For example, assuming $K = 5$, there are 5 possible sets of values for $\{z_{ik}\}$ that satisfy the constraints in Equation (8.6)-(8.8). If we set $\{z_{i,1..3} = 1; z_{i,4} = 0\}$, then x_i is in the

fourth segment, i.e., $x_i \in [0.6, 0.8]$. Equation (8.9) defines the feasible regions for $\boldsymbol{x}$. More details are in Chapter 7.

8.2 COCOMO– A Branch-and-Price Algorithm

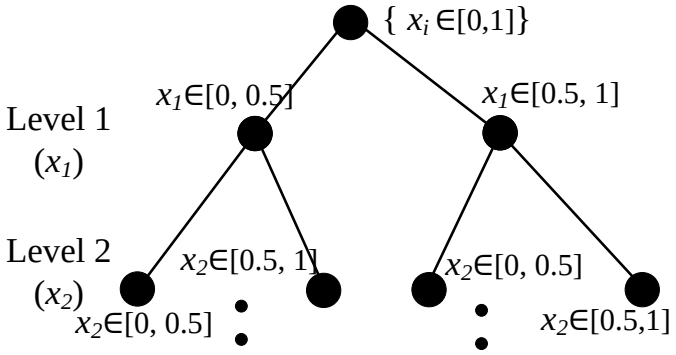


Figure 8.1: Branching Tree

G-PASAQ assumes that the set of defender pure strategies ($\mathcal{A}$) can be explicitly enumerated. In massive SSGs, $\mathcal{A}$ cannot be enumerated; COCOMO (COlumn generation for COmplex adversary MOdels) attempts in such cases to use the branch-and-price approach to scale-up G-PASAQ. COCOMO exploits the fact that the integer variables in G-PASAQ represent the particular piecewise linear segments each marginal x_i belongs to and defines a branching tree shown in Figure 8.1. Initially at the root node, all the integer variables are relaxed to be continuous, indicating that none of the x_i are set to any fixed ranges. The i^{th} level in the tree is associated with marginal x_i. If each marginal is divided into K segments, each node has K children. For example, in Figure 8.1, the two nodes at level 1 are associated with the two possible ranges of marginal x_1: the left node sets $x_1 \in [0, 0.5]$, realized by setting $z_{11} = 0$; the right node sets $x_1 \in [0.5, 1]$, realized by

setting $z_{11} = 1$. As we move deeper, more integer variable values are set. The tree has a depth of $|\mathcal{T}|$ and $K^{|\mathcal{T}|}$ nodes in total.

COCOMO starts from the root node in the branching queue and iterates until the queue is empty. In each iteration, the top node in the branching queue is first branched into a set of children. For each child node, the upper bound (*UB'*) and the lower bound (*LB'*) are estimated. If the two bounds are not close enough, the node is added to the branching queue. COCOMO keeps a record of the best lower bound solution ($\tilde{LB}$) found so far, and uses that to prune all the unvisited nodes in the branching queue. In the end, the defender strategy associated with this best lower bound is returned as the solution.

Upper Bound Estimation: To generate tighter upper bounds, we run G-PASAQ at each node of COCOMO, where the values of some variables z_{ik} are set to either 0 or 1 (see Figure 8.1). We obtain the upper bound by relaxing the rest of the integer variables to be continuous, resulting in an LP called `UpperBound-LP`. `UpperBound-LP` cannot escape the large number of variables a_j and A_j; hence we apply the standard *column generation* technique: we start by solving `UpperBound-LP` with a subset of columns, i.e., defender strategies A_j, and iteratively add more columns with negative *reduced cost*. Let's first rewrite Equation (8.9) based on the definition of $\mathcal{X}_f$ from Equation (8.2) and (8.3).

$$\sum_{k=1..K} x_{ik} - \sum_{A_j \in \mathcal{A}} a_j A_{ij} = 0, \ \forall i \in \mathcal{T} \tag{8.10}$$

$$\sum_{A_j \in \mathcal{A}} a_j = 1, \quad a_j \geq 0, A_j \in \mathcal{A} \tag{8.11}$$

$$\sum_{i \in \mathcal{T}} B_{mi} \sum_{k=1..K} x_{ik} \leq b_m, \ \forall m \tag{8.12}$$

The reduced cost of column A_j is $\omega^T A_j - \rho$, where ω and ρ are the duals of Equation (8.10) and (8.11) respectively. Given the optimal duals of the current iteration of `UpperBound-LP`, a separate *Slave* process provides a new column with the minimum reduced cost; the process iterates until convergence.

Slave: Given the spatio-temporal constraints, the *Slave* can often be formulated as a minimum-cost integer flow problem on a polynomial-sized network, e.g., [Jain et al., 2010a] provide such a *Slave* formulation with application to the FAMS domain. A good example of such formulation can be found in [Jain et al., 2010a] for the FAMS domain: a target is a flight and is represented by a vertex in the network. If two flights can be covered by the same air marshal on the same schedule, there is an edge between the two corresponding vertices. A feasible flow in the network represents a feasible pure strategy of the defender. Similarly here, $\omega^T A_j - \rho$ is assigned as the cost to the vertex representing target i in the network.

More generally, if the set $\mathcal{A}$ of pure strategy vectors A_j that satisfies the spatio-temporal constraints can be formulated as the feasible set of a polynomial number of integer linear constraints: $A_j \in \mathcal{A} = \{\boldsymbol{s} \in \{0,1\}^{|\mathcal{T}|} : C\boldsymbol{s} \leq \boldsymbol{c}\}$, the slave amounts to solving the following integer linear program.

$$\{\min_{\boldsymbol{s}} \ \boldsymbol{s}^T\boldsymbol{y} - \boldsymbol{s}^T B^T \boldsymbol{g} + u \mid s_i \in \{0,1\}, \forall i \in \mathcal{T}; C\boldsymbol{s} \leq \boldsymbol{c}\}$$

Lower Bound Estimation: A subset of the columns will be generated while solving the `UpperBound-LP`. The lower bound of the same node is computed by running G-PASAQ with this subset of columns.

Pruning: COCOMO keeps a record of the best lower bound solution, $\tilde{LB}$, that has been found so far. After branching the current node, pruning is applied to all the unvisited nodes in the

branching queue. The nodes whose upper bound is lower than $\tilde{LB}$ are pruned from the branching queue.

The exponential size of CoCoMo's branching tree ($K^{|\mathcal{T}|}$) and the need to run G-PASAQ for each of branching nodes, ultimately leads to its inefficiency.

8.3 BLADE– A Cutting-Plane Algorithm

Despite our effort for efficiency in CoCoMo, the need to run column generation at each of the $K^{|\mathcal{T}|}$ nodes ultimately leads to its inefficiency. BLADE (Boosted piecewise Linear Approximation of DEfender strategy with arbitrary constraints) uses the cutting-plane approach to scale-up G-PASAQ, and avoids running column generation at each node. Algorithm 6 presents BLADE. The *Master* is a *modified version* of `P1` with a relaxed defender strategy space, defined by the set of boundaries $\tilde{H}\boldsymbol{x} \leq \tilde{\boldsymbol{h}}$. In Line (2), $(\tilde{H}, \tilde{\boldsymbol{h}})$ is initialized with the user-specified constraints, $(B, \boldsymbol{b})$. The solution found by the *Master*, i.e., $\tilde{\boldsymbol{x}}$, provides an upper bound (*UB*) of the solution for `P1`. In each iteration, the *Separation Oracle* checks whether or not $\tilde{\boldsymbol{x}} \in \mathcal{X}_f$. If so, the optimal solution of `P1` has been found; otherwise, a new cutting plane $H_l\boldsymbol{x} \leq h_l$ is returned to further restrict the search space in the *Master*. The *Separation Oracle* also returns a feasible solution $\boldsymbol{x}_f$ 'closest' to the infeasible solution $\tilde{\boldsymbol{x}}$, which provides a lower bound ($LB = F(\boldsymbol{x}_f)$) of the solution for `P1`. In Line (9), we improve our lower bound estimation to further speed up the algorithm. The algorithm terminates when *UB* and *LB* are close enough, i.e., $UB - LB \leq \epsilon$.

Algorithm 5: BLADE

Input: $\{R_i^d, P_i^d, R_i^a, P_i^a\}$, $(B, \boldsymbol{b})$, ϵ;
$(\tilde{H}, \tilde{\boldsymbol{h}}) \leftarrow (B, \boldsymbol{b})$, *feasible* $\leftarrow$ *false*;
$UB \leftarrow M, LB \leftarrow -M$;
while $UB - LB > \epsilon$ **do**
    $(UB, \tilde{\boldsymbol{x}}) \leftarrow$ `Master`$(\tilde{H}, \tilde{h})$;
    (*feasible*, H_l, h_l, $\boldsymbol{x}_f$)$\leftarrow$ `SeparationOracle`$(\tilde{\boldsymbol{x}})$;
    $\tilde{H} \leftarrow \tilde{H} \cup H_l$, $\tilde{\boldsymbol{h}} \leftarrow \tilde{\boldsymbol{h}} \cup h_l$;
    if *feasible* $\neq$ *true* **then**
        $(LB, \boldsymbol{x}_l) \leftarrow$ `LowerBoundEstimator`$(\tilde{H}, \tilde{\boldsymbol{h}})$;
    end
end
return $\boldsymbol{x}_l$;

8.3.1 Master

We first reformulate `P1` by representing its feasible region using the set of bouandries instead of the extreme points:

$$\texttt{P1.1}: \{\max_{\boldsymbol{x}} F(\boldsymbol{x}) \mid H\boldsymbol{x} \leq \boldsymbol{h}; B\boldsymbol{x} \leq \boldsymbol{b}; 0 \leq x_i \leq 1, \forall i \in \mathcal{T}\}$$

H is a N-by-$|\mathcal{T}|$ matrix, where N is the number of linear boundaries of the convex hull. Each row, $H_l\boldsymbol{x} \leq h_l$, represents a linear boundary of $\mathcal{X}_{f1}$. In the presence of user-specified constraints, $B\boldsymbol{x} \leq \boldsymbol{b}$ is added to the boundary set of $\mathcal{X}_f$, as defined in Equation (8.3). However, we cannot directly solve `P1.1` because H and $\boldsymbol{h}$ are not initially given.

In BLADE, the *Master* solves `P1.1` using G-PASAQ with a subset of the boundaries of $\mathcal{X}_f$. More specifically, Equation (8.9) is rewritten as Equation (8.12) and (8.13):

$$\sum_{i \in \mathcal{T}} \tilde{H}_{li} \sum_{k=1..K} x_{ik} \leq \tilde{h}_l, \ \forall l \tag{8.13}$$

$(\tilde{H}, \tilde{\boldsymbol{h}})$ in Equation (8.13) represents the subset of the boundaries for $\mathcal{X}_f$. The solution of *Master*, denoted as $\tilde{\boldsymbol{x}}$, then provides an upper bound on the solution of `P1.1`: $F(\tilde{\boldsymbol{x}}) \geq F(\boldsymbol{x}^*)$, where

$\boldsymbol{x}^*$ denote the optimal solution of `P1.1`. As the algorithm keeps refining the feasible region by adding new boundaries to the *Master*, this upper bound gets tighter.

Given $\tilde{\boldsymbol{x}}$ as the relaxed solution from the *Master*, we check whether it belongs to $\mathcal{X}_f$. If so, we have found the optimal solution of `P1.1`. Otherwise, we further restrict the feasible region in the *Master* via a cut to separate the current infeasible solution and the original feasible region.

8.3.2 Separation Oracle

One standard approach for checking feasibility and generating cutting planes is to apply Farkas' Lemma, as in [Papadimitriou and Roughgarden, 2008]. However, the resulting cutting planes are not guaranteed to be ***deep*** cuts that touch the feasible region and therefore eliminate as much of the infeasible region as possible. Instead, we use a norm-minimization approach for the *Separation Oracle* in BLADE, which efficiently checks the feasibility of $\tilde{\boldsymbol{x}}$, and generates a deep cut to separate $\mathcal{X}_f$ from an infeasible $\tilde{\boldsymbol{x}}$. Additionally, our approach finds a feasible point that is closest to $\tilde{\boldsymbol{x}}$, allowing us to compute a lower bound on the optimal objective.

Check Feasibility: The *Separation Oracle* checks the feasibility of $\tilde{\boldsymbol{x}}$ by minimizing its distance to the feasible region. If the minimum distance is 0, $\tilde{\boldsymbol{x}}$ is within the feasible region. We choose 1-norm to measure the distance between $\tilde{\boldsymbol{x}}$ and any feasible point, as 1-norm leads to a

Linear Program (LP), which allows the use of column generation to deal with large defender strategy space. We first show the `Min-1-Norm` LP in Equation (8.14)-(8.18),

$$\min_{\boldsymbol{a},\boldsymbol{z}} \quad \sum_{i\in\mathcal{T}} z_i \tag{8.14}$$

$$\text{s.t.} \quad \boldsymbol{z} + A\boldsymbol{a} \geq \tilde{\boldsymbol{x}} \tag{8.15}$$

$$\boldsymbol{z} - A\boldsymbol{a} \geq -\tilde{\boldsymbol{x}} \tag{8.16}$$

$$-BA\boldsymbol{a} \geq -\boldsymbol{b} \tag{8.17}$$

$$\sum_{A_j\in\mathcal{A}} a_j = 1, \quad a_j \geq 0, \ \forall A_j \in \mathcal{A} \tag{8.18}$$

In the above LP, a marginal coverage is represented by the set of defender pure strategies: $A\boldsymbol{a}$. Constraint (8.17) and (8.18) enforces that $A\boldsymbol{a}$ satisfies both the spatio-temporal constraints and the user-specified constraints. The 1-norm distance between the given marginal $\tilde{\boldsymbol{x}}$ and $A\boldsymbol{a}$ is represented by vector $\boldsymbol{z}$. This is obtained by combining Constraints (8.15) and (8.16): $-\boldsymbol{z} \leq |A\boldsymbol{a} - \tilde{\boldsymbol{x}}| \leq \boldsymbol{z}$. The objective function minimizes the 1-norm of $\boldsymbol{z}$, therefore the 1-norm distance between $\tilde{\mathrm{x}}$ and any given feasible marginal is minimized.

Lemma 10. *Given a marginal* $\tilde{\mathbf{x}}$*, let* $(\mathbf{z}^*, \mathbf{a}^*)$ *be the optimal solution of the corresponding* `Min-1-Norm` *LP.* $\tilde{\mathbf{x}}$ *is feasible if and only if* $\sum_{i\in\mathcal{T}} z_i^* = 0$*. Furthermore,* $A\mathbf{a}^*$ *provides the feasible marginal with the minimum 1-norm distance to* $\tilde{\mathbf{x}}$*.*

Generate Cut: If $\tilde{\boldsymbol{x}}$ is infeasible, we need to further restrict the relaxed region in the *Master*. Theoretically, any hyperplane that separates $\tilde{\boldsymbol{x}}$ from the feasible region could be used. In practice, a deep cut is preferred. Let $\boldsymbol{w}$, $\boldsymbol{v}$, $\boldsymbol{g}$ and u be the dual variables of Constraints (8.15), (8.16), (8.17) and (8.18) respectively; and let $\boldsymbol{y} = \boldsymbol{w} - \boldsymbol{v}$.

Lemma 11. *Given an infeasible marginal* $\tilde{\mathbf{x}}$*, let* $(\mathbf{y}^*, \mathbf{g}^*, u^*)$ *be the dual values at the optimal solution of the corresponding* `Min-1-Norm` *LP. The hyperplane* $(\mathbf{y}^*)^T\mathbf{x} - (\mathbf{g}^*)^T\mathbf{b} + u^* = 0$ *separates* $\tilde{\mathbf{x}}$ *and* $\mathcal{X}_f$*:*

$$(\mathbf{y}^*)^T\tilde{\mathbf{x}} - (\mathbf{g}^*)^T\mathbf{b} + u^* > 0 \tag{8.19}$$

$$(\mathbf{y}^*)^T\mathbf{x} - (\mathbf{g}^*)^T\mathbf{b} + u^* \leq 0, \quad \forall \mathbf{x} \in \mathcal{X}_f \tag{8.20}$$

Proof. The dual of the `Min-1-Norm` LP is:

$$\max_{\boldsymbol{y},u,\boldsymbol{g}} \quad \tilde{\boldsymbol{x}}^T\boldsymbol{y} - \boldsymbol{b}^T\boldsymbol{g} + u \tag{8.21}$$

$$\text{s.t.} \quad A^T\boldsymbol{y} - A^TB^T\boldsymbol{g} + u \leq 0 \tag{8.22}$$

$$1 \geq \boldsymbol{y} \geq -1, \ \boldsymbol{g} \geq 0 \tag{8.23}$$

Equation (8.19) can be proved using LP duality. Since $\tilde{\boldsymbol{x}}$ is infeasible, the minimum of the corresponding `Min-1-Norm` LP is strictly positive. Therefore, the maximum of the dual LP is also strictly positive.

We now prove the contrapositive of Equation (8.20):

$$(\boldsymbol{y}^*)^T\boldsymbol{x} - (\boldsymbol{g}^*)^T\boldsymbol{b} + u^* > 0 \Rightarrow \boldsymbol{x} \textit{ is not feasible}$$

Given any $\boldsymbol{x}'$, there is a corresponding LP with the same formulation as that in Equation (8.21)-(8.23). Let $(\boldsymbol{y}', u', \boldsymbol{g}')$ be the optimal solution of this LP. Note that, $(\boldsymbol{y}^*, u^*, \boldsymbol{g}^*)$ is a feasible solution of this LP. Therefore,

$$(\boldsymbol{y}')^T\boldsymbol{x} - (\boldsymbol{g}')^T\boldsymbol{b} + u' \geq (\boldsymbol{y}^*)^T\boldsymbol{x} - (\boldsymbol{g}^*)^T\boldsymbol{b} + u^* > 0$$

This indicates that the minimum 1-norm distance between $\boldsymbol{x}$ and $\mathcal{X}_f$ is strictly positive. Hence, $\boldsymbol{x}$ is infeasible. □

Lemma 12. *Equation (8.20) is a* **deep** *cut that touches the feasible convex hull $\mathcal{X}_f$.*

Proof. For simplicity, we consider the cases without user-specified constraints. The cut in Equation (8.20) then becomes $(\boldsymbol{y}^*)^T\boldsymbol{x} + u^* \leq 0$. Let a_j be the dual of the j^{th} constraint in Equation (8.22) and $\boldsymbol{a}^* = \langle a_j^* \rangle$ be the dual at the optimal solution of LP in Equation (8.21)-(8.23). According to the LP duality, $A\boldsymbol{a}^*$ is the optimal solution of the `Min-1-Norm` LP. Therefore, $A\boldsymbol{a}^*$ is the feasible marginal with the minimum 1-Norm distance to $\tilde{\boldsymbol{x}}$. Furthermore, $\forall a_j^* > 0$, the corresponding constraint in Equation (8.22) is active, i.e. $(\boldsymbol{y}^*)^T A_j + u^* = 0$. Hence, the extreme point A_j is on the cutting-plane. □

Therefore, by solving either the `Min-1-Norm` LP or its dual LP, the *Separation Oracle* can not only check the feasibility of a given marginal, but also generate a deep cut. We choose to solve the dual LP in Equation (8.21)-(8.23), since it gives the constraint directly as shown in Equation (8.20). However, since in our case the set of the defender's pure strategies is too large to be enumerated, the constraints of the LP cannot be enumerated.We solve the LP using a constraint generation approach, outlined in Algorithm 7. Specifically, we solve the LP in Equation (8.21)-

Algorithm 6: Separation Oracle

Input: $\{R_i^d, P_i^d, R_i^a, P_i^a\}$, (B, b), $\tilde{\boldsymbol{x}}$, $A^{(0)}$;
$A \leftarrow A^{(0)}$, $A_l \leftarrow A_1$;
while $A_l \neq \emptyset$ **do**
    $A \leftarrow A \cup A_l$;
    $(\boldsymbol{y}^*, u^*, \boldsymbol{g}^*) \leftarrow$ *Solve-Separation-Oracle-LP*(A);
    $A_l \leftarrow$`SecondaryOracle`$(\boldsymbol{y}^*, u^*, \boldsymbol{g}^*)$;
end
return $(\boldsymbol{y}^*, u^*, \boldsymbol{g}^*)$;

(8.23) with a subset of constraints first, and use a *Secondary Oracle* to check whether the relaxed solution violates any of the other constraints.

Secondary Oracle: The secondary oracle is executed at Line (6) in Algorithm 7. If any constraint in Equation (8.22) is violated, the oracle returns the one that is most violated, i.e., A_l with the most negative value of the LHS of Equation (8.22); otherwise, we have found the optimal solution of the LP. The secondary oracle is similar to the *Slave* in CoCoMo.

8.3.3 wBlade

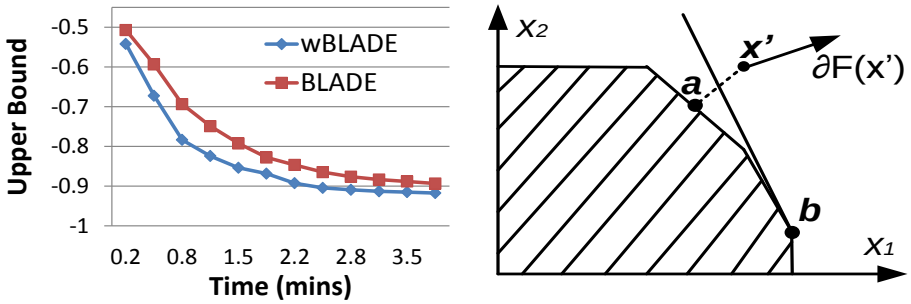
The convergence of Blade depends on how fast the cuts generated by the *Separation Oracle* approximate the feasible set around the optimal solution of `P1.1`. We propose wBlade, which modifies the *Separation Oracle* by changing the norm used to determine the distance to $\mathcal{X}_f$ for one that takes the objective function into account, to bias the cut generated toward the optimal solution. Formally, given the solution $\tilde{\boldsymbol{x}}$ from the *Master*, instead of searching for the feasible point with the 1-norm distance, which is uniform in all dimensions, we modify the objective function of the `Min-1-Norm` LP in Equation (8.14) as:

$$\sum_{i \in \mathcal{T}} (\nabla_i F(\tilde{\boldsymbol{x}}) + \xi) z_i \tag{8.24}$$

where $\nabla_i F(\tilde{\boldsymbol{x}})$ is the gradient of objective function $F(\boldsymbol{x})$ at point $\tilde{\boldsymbol{x}}$ with respect to x_i; ξ is a pre-defined constant to ensure that $\nabla_i F(\tilde{\boldsymbol{x}}) + \xi > 0, \forall i$ so the objective remains a norm. We refer to this modified LP as `Min-Weighted-Norm` LP.

Lemma 13. *A marginal* $\tilde{\mathbf{x}}$ *is feasible if and only if the minimum of the corresponding* `Min-Weighted-Norm` *LP is 0.*

Proof. We already showed that $z_i \geq 0$ represents the absolute difference between $\tilde{\boldsymbol{x}}$ and the feasible point $A\boldsymbol{a}$ on the i^{th} dimension. Combining with $\nabla_i F(\tilde{\boldsymbol{x}}) + \xi > 0, \forall i$, we have $\sum_i (\nabla_i F(\tilde{\boldsymbol{x}}) +$



(a) Upper bounds over time

(b) Weighted 1-norm

Figure 8.2: Minimizing weighted 1-norm distance

$\xi)z_i \geq 0$. According to Lemma 14, $\tilde{\boldsymbol{x}}$ is feasible if and only if the minimum of $\sum_i z_i$ is 0. Hence, if there exists $(\boldsymbol{z}^*, \boldsymbol{a}^*)$ such that $\sum_i z_i^* = 0$, we have $\sum_i (\nabla_i F(\tilde{\boldsymbol{x}}) + \xi) z_i^* = 0$; and vice versa. □

To provide some intuition into why tighter bounds can be obtained by solving `Min-Weighted-Norm` LP, we consider the case when $\nabla_i F(\tilde{\boldsymbol{x}}) > 0$ and $\tilde{\boldsymbol{x}} \geq A\boldsymbol{a}$. First we note that these are typical situations in security games, where having more defense resources tends to benefit the defender. This is the case even if the attacker is boundedly rational, as in the quantal response model. Therefore for most values of $\tilde{\boldsymbol{x}}$ the gradient $\nabla_i F(\tilde{\boldsymbol{x}})$ will be positive. As a result, a solution $\tilde{\boldsymbol{x}}$ of the relaxed problem solved by the master will tend to use more resources than what is feasible, i.e., $\tilde{\boldsymbol{x}} \geq A\boldsymbol{a}$. These properties are confirmed in our numerical experiments.

Then, if we have $\nabla_i F(\tilde{\boldsymbol{x}}) > 0$ and $\tilde{\boldsymbol{x}} \geq A\boldsymbol{a}$, the `Min-Weighted-Norm` LP is equivalent to minimizing $\nabla F(\tilde{\boldsymbol{x}}) \cdot (\tilde{\boldsymbol{x}} - A\boldsymbol{a})$ and hence also to maximizing

$$F(\tilde{\boldsymbol{x}}) + \nabla F(\tilde{\boldsymbol{x}})(A\boldsymbol{a} - \tilde{\boldsymbol{x}}) \tag{8.25}$$

Equation (8.25) is the first-order Taylor approximation of $F(\boldsymbol{x})$, maximizing which should provide a good lower bound if $\tilde{\boldsymbol{x}}$ is close to the feasible region.

Regarding the cuts generated, since $\nabla F(\tilde{\boldsymbol{x}}) > 0$ we can take $\xi = 0$. In this case the `Min-Weighted-Norm` LP is looking for the projection point in $\mathcal{X}_f$ on the highest level-set

perpendicular to $\nabla F(\tilde{\boldsymbol{x}})$. The cut generated, therefore, will be this highest level-set perpendicular to $\nabla F(\tilde{\boldsymbol{x}})$. Since the gradient points in the direction of maximum increase of its function, for sufficiently smooth functions and sufficiently close projection point, points with higher function values than the projection point would be eliminated by this cut. Figure 8.2(b) demonstrates an example: the shaded polyhedron represents the feasible region $\mathcal{X}_f$. Given the infeasible point $\boldsymbol{x}'$ and its gradient $\nabla F(\boldsymbol{x}')$, the solid line perpendicular to $\nabla F(\boldsymbol{x}')$ approximates the level-set of $F(\boldsymbol{x})$. Therefore, the half-space on the direction of $\nabla F(\boldsymbol{x}')$ of that line would tend to have points with higher value of $F(\boldsymbol{x})$ than the other half-space. This suggests that such a cut would prune more infeasible points with high objectives, leading to a tighter upper bound. Confirming this intuition, Figure 8.2(a) shows that over 30 random samples the upper-bound decreases faster in WBLADE: x-axis marks iterations in time, y-axis plots the upper bound.

8.3.4 Quality and Runtime Trade-off

The feasible point returned by the *Separation Oracle* provides a lower bound on `P1.1`. A better lower bound can be achieved by solving a restricted version of `P1.1` (Line 9 in Algorithm 6). This can be done by replacing $\mathcal{X}_{f1}$ with the convex hull formed by the subset of defender pure strategies generated in solving the *Separation Oracle*. These upper and lower bounds allow us to trade off between solution quality and runtime by controlling the threshold ϵ: as soon as $UB - LB \leq \epsilon$ the algorithm returns the feasible solution associated with *LB*, which is guaranteed to be within ϵ of the optimal objective.

8.4 Experimental results

In this section, we compare COCOMO and BLADE assuming two different bounded rationality models. We take FAMS as our example domain. For each setup, we tried 30 game instances. In each game, payoffs R_i^d and R_i^a are random integers from 1 to 10, while P_i^d and P_i^a are random integers from -10 to -1; the feasible schedules for each unit of resources are generated by randomly selecting 2 targets for each schedule (we assume that each air marshal can cover 2 flights on a single trip, similar to [Jain et al., 2010a]). In all experiments, the deployment-to-saturation (d:s) ratio is set to 0.5, which is shown to be computationally harder than any other d:s ratio [Jain et al., 2012]. Furthermore, we set the number of piecewise linear segments to be 15 for each $f_i(x_i)$, given that 10 segments provide a sufficiently good approximation [Yang et al., 2012b]. The results were obtained using CPLEX v12.2 on a standard 2.8GHz machine with 4GB main memory.

The BEST BLADE Given the two versions of BLADE, one with the non-weighted *Separation Oracle* and another with the weighted *Separation Oracle*, we investigate whether it would be more effective to combine them such that two cuts are generated in each iteration. While this combined version CBLADE could generate more cuts per iteration reducing the total number of iterations, the runtime of each iteration might be longer. Our first set of experiment investigates the efficiency of the three BLADE algorithms. Figure 8.3 shows the average runtime of these three algorithms with different number of targets. WBLADE achieves the shortest runtime, as shown in Figure 8.3. Furthermore, although on average CBLADE takes less iterations to converge, it generates more cuts than both BLADE and WBLADE. For example, with 60 targets, CBLADE takes 17 iterations on average to converge, while WBLADE and BLADE take 23 and 29 iterations

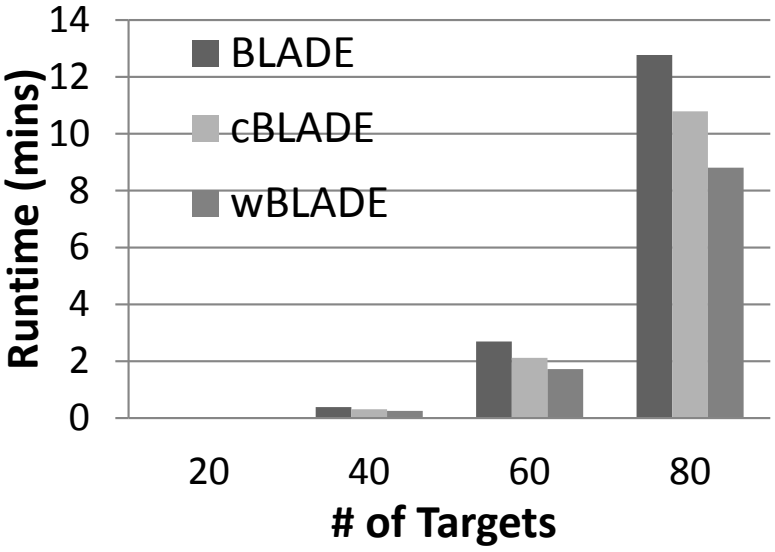
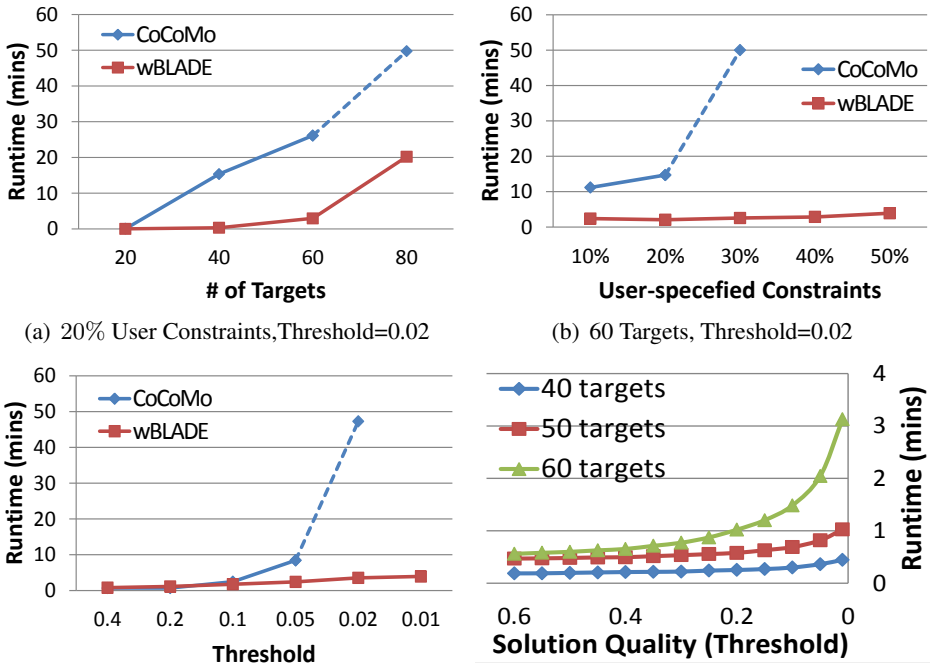


Figure 8.3: Runtime Comparison of the BLADE family

respectively. However, the total cuts generated by CBLADE is 34 (2 cuts per iteration) which is more than WBLADE and BLADE. Given this result, we will use WBLADE as the representative of the BLADE family in the rest of the experiments.

Quantal Response Model Figure 8.4(a), 8.4(b) and 8.4(c) present the average runtime of COCOMO and WBLADE assuming a QR model of the adversary with λ parameter set to 0.76. In the experiment, we set 50 minutes as the runtime limit. The dashed line in the figures indicates that at that point at least some game instances were not completed within this time limit; and the absence of any markers afterward implies the trend continues. COCOMO cannot scale to 80 targets, as shown in Figure 8.4(a). In Figure 8.4(b), we vary the amount of user-specified constraints (i.e. percentage of the number of targets) while fixing the number of targets to be 60. The constraints were randomly generated inequalities of the marginal coverage vector $\langle \boldsymbol{x} \rangle$. Increasing the amount of user-specified constraints doesn't impact the runtime of WBLADE, but



(a) 20% User Constraints,Threshold=0.02

(b) 60 Targets, Threshold=0.02

(c) 60 Targets, 20% User Constraints

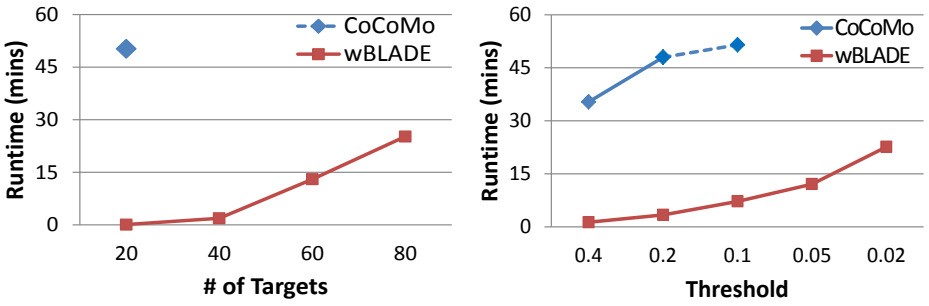
(d) Runtime vs Solution Quality

Figure 8.4: Comparing CoCoMo and BLADE, QR Model

significantly slows down CoCoMo. We then vary the threshold from 0.4 to 0.01 as shown in Figure 8.4(c). CoCoMo is only able to converge when the threshold is larger than 0.05;in comparison, the runtime of wBLADE slowly increases as the threshold decreases. Thus, wBLADE obtain much better solution quality within significantly shorter amount of time than CoCoMo.

We further investigate the tradeoff between solution quality and runtime of wBLADE and show the result in Figure 8.4(d). We gradually increase the solution quality by decreasing the threshold under different number of targets, illustrating runtime-quality tradeoff.

A More Complex Bounded Rationality Model We now set $f_i(x_i) = \frac{1}{1+e^{-\lambda_i x_i}}$, a more complex model than QR.We investigate the impact of model complexity on the runtime of CoCoMo and wBLADE. Figure 8.5(a) and 8.5(b) display the runtime comparison of CoCoMo and wBLADE. As shown in Figure 8.5(a), while CoCoMo could not finish running within 50



(a) 20% User Constraints, Threshold=0.02 (b) 60 Targets, 20% User Constraints

Figure 8.5: Runtime Comparison, QR-Sigmoid model

minutes in any of the settings, the runtime of wBLADE was less than 7 seconds for 20 targets. In Figure 8.5(b), we show that the runtime of BLADE gradually increases as the threshold decreases. In comparison, COCOMO is only able to finish running when the threshold is sufficiently large (≥ 0.4) leading to poor solution quality. Thus, as the bounded rationality model becomes more complex, BLADE's advantage over COCOMO is further magnified.

Chapter 9: Adaptive Resource Allocation and its Application to Wildlife Protection

In many domains, adversary events occur often and generate significant amounts of collectible event data. These domains present new research challenges and opportunities related to learning behavioral models from collected poaching data. One example of such domains is wildlife protection. Illegal poaching is an international problem that leads to the extinction of species and the destruction of ecosystems. As evidenced by dangerously dwindling populations of endangered species, existing anti-poaching mechanisms are insufficient. Compared to the counter-terrorism domain, wildlife crime is an important domain that promotes a wide range of new deployments. In this chapter, I introduce the Protection Assistant for Wildlife Security (PAWS) application - a joint deployment effort done with researchers at Uganda's Queen Elizabeth National Park (QENP) with the goal of improving wildlife ranger patrols.

9.1 Domain

The goal of PAWS is to help conservation agencies improve patrol efficiency such that poachers, from fear of being caught, are deterred from poaching in QENP. Wire snaring is one of the main techniques used by poachers in Africa, including QENP, (as shown in figures 9.1(a),9.1(b));

poachers can set and leave snares unattended, and come back when they think an animal has been captured. In addition, poachers can conduct surveillance on rangers' activities and patrol patterns; wildlife rangers are well-aware that some neighboring villagers will inform poachers of when they leave for patrol and where they are patrolling [Moreto, 2013]. For any number of reasons, such as changes that impact animal migration habits, rangers may change their patrolling patterns; poachers, in turn, continually conduct surveillance on the rangers' changing patrol strategy and adapt their poaching strategies accordingly. As the law enforcement officers of the park, park rangers' primary objective is to stop poaching, and their main method of doing so is to patrol the park. During a patrol, rangers will search for signs of illegal activity inside the park, confiscate any poaching equipment found, and apprehend any persons inside the park illegally (e.g., poachers).

In addition to their normal patrol duties, rangers will collect data on any observed or suspected illegal activity. In most cases, if rangers find wire snares, they will not find the poacher that set them. If the rangers do encounter and apprehend poachers, however, they are sometimes able to make the poachers confess to where they set their snares. After the rangers return to the outpost, collected data is uploaded and analyzed. Eventually, enough data will be collected so that the ranger patrol strategies can be continually updated based on any emerging trends. If snares are found by a ranger patrol, they are recorded as data points. Since it is unknown who placed the snares, we refer to these data points as anonymous data points. Identified data points, when a poacher is captured and divulges where they placed snares, are inherently more useful as they can be used to obtain a more complete behavioral model that can better predict where future poachers will place their traps.

(a) A lioness caught in a snare. (b) A caught poacher holding up a snare.

Figure 9.1: Lioness photo courtesy of John Coppinger, Remote Africa Safaris Ltd. Poacher snare photo taken by Andrew Lemieux.

For this deployment, a poacher placing a snare in an area represents an attack. In order to have a tractable space for computing defender strategies, we discretize areas into a grid where each cell represents 1 square kilometer, and every cell in the grid could contain wildlife and is thus a valid target for attackers. Terrain also has an impact; poachers and rangers can travel further if they are traversing grasslands instead of a dense forest of varying elevations. In order to simplify distance calculations in our model, we currently focus on one type of terrain, grasslands. Future work will focus on incorporating different types of terrain into the model. Areas of high animal density, such as areas that contain fresh water (e.g., watering holes, lakes), are known to be high-risk areas for poaching [Wato et al., 2006; Moreto, 2013; Montesh, 2013]. Distance is also an important factor; a snare density study demonstrated that the density began to decrease significantly once they began travelling more than 4 kilometers away from the international border [Wato et al., 2006]. This finding is intuitive as poachers need to carry back any poached animals or goods, and longer distances will increase the chances of spoilage and apprehension. Even for Ugandan poachers, distance travelled will still be a factor based on similar concerns. Despite the available information from these studies, there are still too many areas for rangers to patrol, and it is a huge cognitive burden to account for these factors (in addition to physical distance constraints

and available rangers) while constantly creating new, unpredictable patrols. Based on all of these factors, PAWS will aid patrol managers and determine an optimal strategy that will enable park rangers to effectively cover these numerous areas with their limited resources.

9.2 Model in PAWS

9.2.1 Stackelberg Game Formulation

Based on our discussion of the wildlife crime domain and its various parameters of interest, we apply a game theoretic framework, more specifically Stackelberg Security Games (SSGs), to the problem and first model the interaction between the rangers and the poachers. In a SSG, there are two types of players: the defender (leader) commits to a strategy first; the follower then responds after observing the leader's strategy. The defender's goal is to protect a set of targets, with limited security resources, from being attacked by the adversary. The adversary will first conduct surveillance to learn about the defender's strategy, and then he (he by convention) will select a target to attack.

In the wildlife crime problem, the ranger plays as the leader and the poachers are the followers. While the rangers are trying to protect animals by patrolling locations where they frequently appear, the poachers are trying to poach the animals at these areas. As discussed earlier, we discretize the area into a grid where each cell represents 1 square kilometer. We use $\mathcal{T}$ to denote the set of locations that can be targeted by the poacher, where $i \in \mathcal{T}$ represents the i^{th} target. If the poacher selects target i and it is covered by the rangers, he receives a utility of $U^c_{p,i}$. If the selected target is not covered by rangers, he receives a utility of $U^u_{p,i}$. The ranger's utility is denoted

similarly by $U^c_{r,i}$ and $U^u_{r,i}$. As a key property of SSG, we assume $U^c_{p,i} \leq U^u_{p,i}$ and $U^c_{r,i} \geq U^u_{r,i}$. Simply put, adding resources to cover a target hurts poachers and helps the rangers.

As discussed in the previous section 9.1, animal density is a key factor in determining poaching risk, and we thus model it as the primary determinant of reward for poachers (i.e., $U^u_{p,i}$) and penalty for rangers (i.e., $U^u_{r,i}$). Areas with a high density of animals are attractive to poachers since they are more likely to have a successful hunt. Similarly, rangers will view these areas as costly if left unprotected. Distance is also a determining factor in poaching reward. Although a poacher may view an area with a large density of animals as attractive, it may be too far away to be rewarding. We also need to model the penalty for poachers (i.e., $U^c_{p,i}$) and reward for rangers (i.e., $U^c_{r,i}$). If the poachers attack a defended area, they will incur a fixed penalty that represents a fine. The poachers will also incur an additional penalty that increases with the distance that they travel from their starting point. Rangers will receive a flat (i.e., uniform) reward based on the poacher's fixed penalty but not on the distance travelled. This uniform reward represents the ranger's lack of preference on where or how poachers are found; as long as poachers are apprehended, the patrol is considered a success.

In our SSG model for this wildlife crime problem, we assume a single leader (i.e., a single group of rangers who are executing the same patrolling strategy) and a population of poachers. We also assume that poachers respond to the rangers' patrolling strategy independently, and we defer to future work to consider potential collaboration between poachers. We adopt a compact representation of the rangers' patrolling strategy: $\boldsymbol{x} = \langle x_i \rangle$ where x_i denotes the probability that i will be covered by the rangers. The actual patrol can be derived from this compact representation using sampling techniques similar to those in previous SSG applications [Shieh et al., 2012; Tsai et al., 2009]. Given a defender strategy $\boldsymbol{x}$, we denote the response of a poacher as $\langle q_i(\omega|\boldsymbol{x}) \rangle$,

Table 9.1: Notations used in this paper

$\mathcal{T}$	Set of targets; $i \in \mathcal{T}$ denotes target i
x_i	Probability that target i is covered by a resource
$U^c_{r,i}$	Ranger utility for covering i if it's selected by the poacher
$U^u_{r,i}$	Ranger utility for not covering i if it's selected
$U^c_{p,i}$	Poacher utility for selecting i if it's covered
$U^u_{p,i}$	Poacher utility for selecting i if it's not covered
ω	Parameter of the SUQR model
$f(\omega)$	Probability density function of ω
$U_r(\boldsymbol{x}\|\omega)$	Ranger expected utility by playing strategy $\boldsymbol{x}$ against a poach with the model parameter ω
$U_r(\boldsymbol{x})$	Ranger expected utility by playing strategy $\boldsymbol{x}$ against the whole population of the poachers
$q_i(\omega\|\mathcal{G})$	Probability that poacher with parameter ω selects target i in game $\mathcal{G}$

where $q_i(\omega|\boldsymbol{x})$ represents the probability that the poacher will select target i. The parameter ω is associated with the poacher's behavioral model, which we will discuss in more details in Section 9.2.2. Table 9.1 lists key notations used in this paper.

We model the repeated crime activities of the poachers as the following: in each round of the interaction between the rangers and the poachers, the ranger executes the same mixed strategy over a period of time (e.g., a month); the poachers will first conduct surveillance on the rangers' patrolling strategy and then respond. If the ranger switches the patrolling strategy, a new round starts. We assume that the poachers are myopic (i.e., they make their decision based on their knowledge of the ranger's strategy in the current round). In this paper, we also assume the poachers' surveillance grants them perfect knowledge about the rangers' strategy; we defer to future work to consider the noise in poachers' understanding of the rangers' strategy due to limited observations.

9.2.2 Behavioral Heterogeneity

The model we use to predict the behavior of the poachers is based on the SUQR model as described in Chapter 5 and replaces the assumption of a single parameter setting with a probabilistic distribution of the model parameter in order to incorporate the heterogeneity among a large population of adversaries. SUQR extends the classic quantal response model by replacing the expected utility function with a subjective utility function:

$$SU_i(\omega) = \omega_1 x_i + \omega_2 U^u_{p,i} + \omega_3 U^c_{p,i} \tag{9.1}$$

where the parameter $\omega = \langle \omega_1, \omega_2, \omega_3 \rangle$ measures the weight of each factor in the adversary's decision making process. In Chapter 5, ω was learned using data collected with human subjects from Amazon Mechanical Turk and assumed that there was a single parameter ω. We will show that the parameters learned for individuals in the data set differ from each other. We then show that the model's predictive power significantly improves if the parameter is changed from a single value to a probabilistic distribution.

In the data set collected with the subjects from Amazon Mechanical Turk, each subject played 25-30 games. In total, data was collected on about 760 subjects. We learn the SUQR parameter for each individual by maximizing the log-likelihood defined in Equation (9.2)

$$\log L(\omega) = \sum_k \log(q_{c_k}(\omega|\mathcal{G}_k)) \tag{9.2}$$

where, $\mathcal{G}_k$ denotes the k^{th} game played by the subject. c_k is the index of the target selected by the subject in this game. q_{c_k} represents the probability that target c_k will be selected by the subject predicted by the SUQR model, which is computed as the following:

$$q_{c_k}(\omega|\mathcal{G}_k) = \frac{e^{SU_{c_k}(\omega|\mathcal{G}_k)}}{\sum_i e^{SU_i(\omega|\mathcal{G}_k)}} \tag{9.3}$$

Table 9.2: Log-likelihood

	Single Parameter Setting	Parameter Learned for each subject
Training Set	-1.62	-1.09
Testing Set	-1.68	-1.15

where, $SU_i(\omega|\mathcal{G}_k)$ is the subjective utility function as described in Equation (9.1) given a game instance $\mathcal{G}_k$. Figure 9.2 displays the empirical PDF of ω. It shows a shape of normal distribution in all three dimensions. Furthermore, we report in Table 9.2 the average log-likelihood of the SUQR model with the parameter value learned for each subject. We also include in Table 9.2 the log-likelihood of the SUQR model with the assumption that the parameter value is the same for all the subjects. The results are evaluated using cross-validation. Table 9.2 shows that the predictive power of the model improves by tuning the parameter for each subject, since the log-likelihood of the prediction by the model is increased. On average, the log-likelihood of the SUQR model with the parameter learned for each subject is 0.53 higher than that with a uniform parameter across all subjects. In other words, the prediction of the former model is 1.70 (i.e. $e^{0.53}$) times more likely than that of the latter.

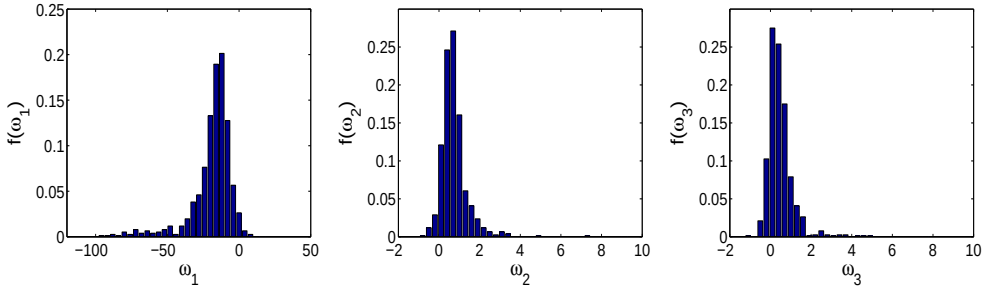


Figure 9.2: Empirical Marginal PDF of the SUQR parameter among all the 760 subjects

Given the results shown in Figure 9.2, we assume a probabilistic, normal distribution of the SUQR parameter ω in order to incorporate the heterogeneity of the decision-making process

of the whole population of poachers. The SUQR model with a specific value of ω essentially represents one type of poacher. With the continuous distribution of ω, we are indeed facing a Bayesian Stackelberg game with infinite types. We denote the probability density function of ω as $f(\omega)$.

9.2.3 Adapting Patrolling Strategy using Historical Crime Data

In the domain of wildlife crime, if the behavioral model of the whole adversary population is given, the optimal patrolling strategy $\boldsymbol{x}^*$ is the one that maximizes the expected utility of the rangers.

$$\boldsymbol{x}^* = arg\max_{\boldsymbol{x}} \oint U_r(\boldsymbol{x}|\omega) f(\omega) d\omega \tag{9.4}$$

where, $U_r(\boldsymbol{x}|\omega)$ is the rangers' expected utility by executing strategy x against a poacher that has a model parameter of ω. $U_r(\boldsymbol{x}|\omega)$ is computed as the following,

$$U_r(\boldsymbol{x}|\omega) = \sum_i U_{r,i}(\boldsymbol{x}|\omega) q_i(\omega|\mathcal{G}) \tag{9.5}$$

where, $U_{r,i}(\boldsymbol{x}|\omega)$ is the rangers' expected utility if target i is selected by the poachers. $U_r(\boldsymbol{x}|\omega)$ is a nonlinear fractional function given that $q_i(\omega|\mathcal{G})$ follows the prediction of the SUQR model.

In reality, the behavioral model of the adversary population is unknown to the rangers. Thus, a key challenge for obtaining an optimal patrolling strategy is to learn the poachers' behavioral models. More specifically, we want to learn the distribution of the SUQR model parameter. In the wildlife crime problem, data is often available about historical crime activities. Recall that these data points record the snares found by the rangers, which can be either anonymous or identified. The identified crime data that is linked to an individual poacher can be used to learn his behavioral model (i.e., estimate the SUQR model parameter for that poacher). In contrast, it is impossible

to directly use anonymous crime data to build a behavioral model for any individuals. In theory, with enough identified crime data, we could estimate the underlying population distribution of ω directly. In reality, however, identified crime data is rare compared to anonymous crime data.

9.3 Research Advances in PAWS

Recall that existing techniques in SSG cannot be applied directly to PAWS due to the new challenges coming from this new domain. In this section, we describe the novel research advances developed for solving the SSG in PAWS.

9.3.1 Learn the Behavioral Model

At the beginning of the game, rangers only know that the distribution of the poacher population's model parameter follows a normal distribution. The goal is to learn the multi-variable normal distribution (i.e., the mean μ and the covariance matrix Σ) of the 3-dimensional SUQR model parameter ω as data becomes available.

As previously discussed, identified data, although sparse, can be used to directly learn poachers' individual behavioral models. Since it is sparse, it takes a much longer time to collect enough data to learn a reasonable distribution. In contrast, there is much more anonymous crime data collected. As we will show, we can learn the behavioral model of the poacher population using these two types of data. Furthermore, we combine the use of the sparse identified data to boost the convergence of the learning.

Let's first define the format of the data collected in each round of the game. Let $N_a^{(t)}$ be the number of anonymous crimes observed by the ranger in round t and $N_c^{(t)}$ be the number of

captured poachers in round t. Furthermore, let $\mathcal{A}^{(t)} = \{a_j^{(t)} | j = 1, ..., N_a^{(t)}\}$ denote the set of targets chosen by the anonymous poachers in round t and $\Omega^{(t)} = \{\omega_k^{(t)} | k = 1, ..., N_c^{(t)}\}$ denote the set of parameter values associated with the captured poachers in round t. $\omega_k^{(t)}$ is the SUQR parameter of the k^{th} captured poacher in round t. We assume that a captured poacher will confess his entire crime history in all previous rounds. For the k^{th} captured poacher in round t, we denote $\mathcal{C}_k^{(t)} = \{c_{k,l}^{(t)}\}$ as the set of crimes committed by him, where the index l in $c_{k,l}^{(t)}$ represents the l^{th} crime committed by him. $c_{k,l}^{(t)} = (\alpha_{k,l}^{(t)}, \boldsymbol{x}_{k,l}^{(t)})$ includes the target chosen by the poacher when the crime was committed (denoted as $\alpha_{k,l}^{(t)}$) and the resource allocation strategy of the rangers at the time (denoted as $\boldsymbol{x}_{k,l}^{(t)}$). To simplify the notation, we denote $\alpha_{k,l}^{(t)}$ as α_l and $\boldsymbol{x}_{k,l}^{(t)}$ as χ_l in the following part of the paper.

9.3.1.1 Learning with the Identified Data

For each captured poacher, the associated SUQR model parameter can be estimated with Maximum Likelihood Estimation (MLE).

$$\begin{aligned} \omega_k^{(t)} &= arg \max_{\omega} \log L(\omega | \mathcal{C}_k^{(t)}) \\ &= arg \max_{\omega} \sum_{l} \log(q_{\alpha_l}(\omega | \chi_l)) \end{aligned} \tag{9.6}$$

where, $q_{\alpha_l}(\omega|\chi_l)$ is the predicted probability that the k^{th} captured poacher chooses target α_l when he committed the crime after observing χ_l as the resource allocation strategy of the rangers. It can be shown that $\log L(\omega|\mathcal{C}_k^{(t)})$ is a concave function, since the Hessian matrix is negative semi-definite.

At round t, there are in total $\sum_{\tau=1}^{t} N_c^{(\tau)}$ poachers captured. After learning the model parameter ω for each of these poachers, there are $\sum_{\tau=1}^{t} N_c^{(\tau)}$ data samples collected from the distribution

of the poacher population. By applying MLE, the distribution of ω can be learned from these data samples. Given that ω follows a 3-dimensional normal distribution, the mean and the covariance matrix learned with MLE is calculated as the following:

$$\mu^{(t)} = \frac{1}{\sum_{\tau=1}^{t} N_c^{(\tau)}} \sum_{\tau=1}^{t} \sum_{\omega \in \Omega^{(\tau)}} \omega \tag{9.7}$$

$$\Sigma^{(t)} = \frac{1}{\sum_{\tau=1}^{t} N_c^{(\tau)}} \sum_{\tau=1}^{t} \sum_{\omega \in \Omega^{(\tau)}} (\omega - \mu^{(t)})(\omega - \mu^{(t)})^T \tag{9.8}$$

9.3.1.2 Learning with the Anonymous Data

Each anonymous data item records the target selected by the poacher. Since no information is recorded about the individual poacher who committed the crime, it is impossible to estimate the model parameter like is done with identified data. One potential approach is to treat each anonymous data point as committed by different independent poachers.

$$\omega_j^{(t)} = arg\max_{\omega} \log L(\omega | a_j^{(t)}, \boldsymbol{x}^{(t)}) \tag{9.9}$$

where, $\boldsymbol{x}^{(t)}$ is the strategy of the rangers in round t. $\omega_j^{(t)}$ denotes the estimated model parameter of the anonymous poacher who committed the j^{th} crime in round t. Note that in each round, the log-likelihood of any given value of ω only depends on the target that was selected by the poacher. Different poachers with different model parameters will be treated the same if they choose the same target in the same round. Let $\tilde{\Omega}^{(t)} = \{\omega_j^{(t)} | j = 1, ..., N_a^{(t)}\}$ represent the set of estimated model parameters associated with the $N_a^{(t)}$ anonymous crimes recorded in round t. Similar to

Algorithm 7: PAWS-Learn

1 Input: $t, \mathcal{C}^{(\tau)}, \mathcal{A}^{(\tau)}, \boldsymbol{x}^{(\tau)}, \forall \tau = 1 \dots t-1$;
2 $(\mu^{(t)}, \Sigma^{(t)}) \leftarrow \texttt{Learn}(\{\mathcal{C}^{(\tau)}, \tau = 1, ..., t-1\})$;
3 $(\{\omega_n\}) \leftarrow \texttt{Sample}(\mu_{(t)}, \Sigma_{(t)}, N_s)$; ;
4 $\pi_n^o \leftarrow \frac{f(\omega_n|\mu^{(t)}, \Sigma^{(t)})}{\sum_{n'} f(\omega_{n'}|\mu^{(t)}, \Sigma^{(t)})}, \forall n$;
5 $\langle \pi_n \rangle \leftarrow \texttt{Refine}(\{\mathcal{C}^{(\tau)}, \tau = 1, ..., t-1\}, \langle \pi_n^o \rangle)$; ;
6 Return $(\{\omega_n\}, \langle \pi_n \rangle)$;

how the identified data was used, the maximum likelihood estimation of the mean and covariance matrix of the distribution of the model parameter can be computed as:

$$\tilde{\mu}^{(t)} = \frac{1}{\sum_{\tau=1}^{t} N_a^{(\tau)}} \sum_{\tau=1}^{t} \sum_{\omega \in \tilde{\Omega}^{(\tau)}} \omega \tag{9.10}$$

$$\tilde{\Sigma}^{(t)} = \frac{1}{\sum_{\tau=1}^{t} N_a^{(\tau)}} \sum_{\tau=1}^{t} \sum_{\omega \in \tilde{\Omega}^{(\tau)}} (\omega - \tilde{\mu}^{(t)})(\omega - \tilde{\mu}^{(t)})^T \tag{9.11}$$

9.3.1.3 Combining the Two Kinds of Data

Identified data provides an accurate measurement of an individual poacher's behavior. However, it leads to slow learning convergence for the population's behavioral model due to its sparseness. While anonymous data provides a noisy estimation of an individual poacher's behavioral model, it gives a sufficiently accurate measurement of the crime distribution of the poacher population due to the large amount of data points. We propose PAWS-Learn, an algorithm to improve the estimation of the model parameter by combining both the identified data and the anonymous data. Algorithm 8 shows the outline of PAWS-Learn.

At round t, PAWS-Learn first uses the identified data to learn the mean and the covariance, as shown in Line (2). It then measures the accuracy of this estimation using the mean square error (MSE) of the predicted crime distribution recorded by the anonymous data.

$$MSE^{(t)}(\mu^{(t)}, \Sigma^{(t)}) = \sum_{i \in \mathcal{T}} (\bar{q}_i(\boldsymbol{x}^{(t)}|\mu^{(t)}, \Sigma^{(t)}) - y_i^{(t)})^2 \tag{9.12}$$

where $y_i^{(t)}$ is the proportion of crimes found at target i as recorded by the anonymous data[1]. $\bar{q}_i(\boldsymbol{x}|\mu^{(t)}, \Sigma^{(t)})$ is the predicted probability that target i will be selected by the poacher population, given $\mathcal{N}(\mu^{(t)}, \Sigma^{(t)})$. Ideally, $\bar{q}_i(\boldsymbol{x}|\mu^{(t)}, \Sigma^{(t)})$ is calculated as

$$\bar{q}_i(\boldsymbol{x}|\mu^{(t)}, \Sigma^{(t)}) = \oint_{\Omega} q_i(\omega, \boldsymbol{x}^{(t)}) f(\omega|\mu^{(t)}, \Sigma^{(t)}) d\omega$$

Let $\pi = \langle \pi_n \rangle$ denote the vector of probabilities associated with the sampled parameter values, where $\sum_n \pi_n = 1$ due to normalization. The predicted probability that target i will be selected by the poacher population in round t is approximated as

$$\bar{q}_i(\boldsymbol{x}^{(t)}) = \sum_n \pi_n q_i(\omega_n, \boldsymbol{x}^{(t)})$$

The quadratic program formulation for minimizing the MSE of the observed crime distribution is shown in Equations (9.13)-(9.15).

$$\min_{\pi} \sum_{i \in \mathcal{T}} (\sum_n \pi_n q_i(\omega_n, \boldsymbol{x}^{(t)}) - y_i^{(t)})^2 \tag{9.13}$$

$$s.t. \sum_n \pi_n = 1, \quad \pi_n \in [0, 1], \forall n \tag{9.14}$$

$$|\pi_n - \pi_n^o| \leq \beta \pi_n^o, \forall n \tag{9.15}$$

Equation (9.15) is to ensure the smoothness of $\langle \pi_n \rangle$ since the values are essentially samples from the probability density function of a normal distribution. More specifically, it constrains π_n to be within a certain distance of the initial value π_n^o. The parameter β is set to decide the range of π_n proportion to π_n^o. As shown in Line (4),π_n^o is set to the pdf of the current estimated distribution $\mathcal{N}(\mu^{(t)}, \Sigma^{(t)})$: $\pi_n^o = C \cdot f(\omega_n|\mu^{(t)}, \Sigma^{(t)})$, where $C = \frac{1}{\sum_n f(\omega_n|\mu^{(t)}, \Sigma^{(t)})}$ is the constant to make sure that $\sum_n \pi_n = 1$. As shown in Line (5), PAWS-Learn refines the probabilities of the sampled parameter values by solving the above quadratic programming problem.

[1]PAWS-Learn currently assumes that in the anonymous data collected by rangers in each round, the observed crime distribution is close to the true distribution.

Algorithm 8: PAWS-Adapt

Input: $N_c^{(t)}$,$N_a^{(t)}$;
$\boldsymbol{x}^{(1)} \leftarrow$ MAXIMIN;
for $\tau = 1, \ldots$ **do**
  $(\mathcal{C}^{(\tau)}, \mathcal{A}^{(\tau)}) \leftarrow \texttt{CollectData}(\boldsymbol{x}^{(\tau)})$;
  $(\{\omega_n\}, \{\pi_n\}) \leftarrow \text{PAWS} - Learn(\mathcal{C}^{(\tau)}, \mathcal{A}^{(\tau)}, \boldsymbol{x}^{(\tau)})$;
  $\boldsymbol{x}^{(\tau+1)} \leftarrow \texttt{ComputeStrategy}(\{\omega_n\}, \{\pi_n\})$
end

9.3.2 Adapting Patrolling Strategy

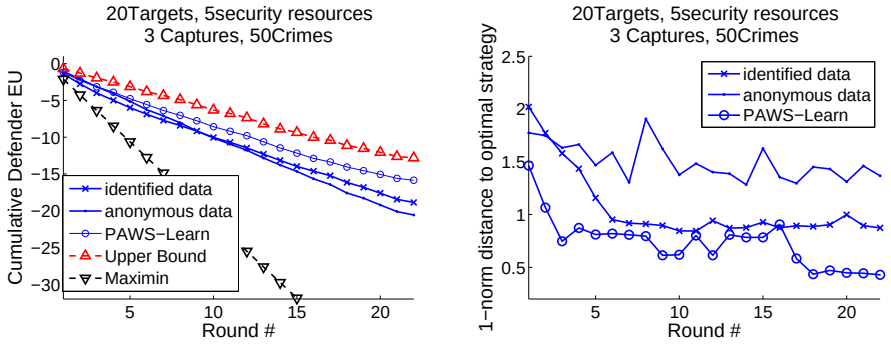
We propose PAWS-Adapt, a framework to adaptively design the patrolling strategy for the rangers. Let $(\mathcal{C}^o, \mathcal{A}^o, \boldsymbol{x}^o)$ be the initial data set. At round t, PAWS-Adapt first estimates the behavioral model with all the historical data by calling PAWS-Learn. Let $(\omega_n^{(t)}, \langle \pi_n^{(t)})$ be the learning results of the poacher population's behaviorial model by PAWS-Learn. PAWS-Adapt then computes the optimal patrolling strategy, based on the current learning result, to execute in the next round.

In computing the optimal patrolling strategy under the given behavioral model, we need to solve the optimization problem in Equation (9.4), which is equivalent to a Bayesian Stackelberg Game with infinite types. With the representation of discretized samples, we approximate the infinite types with a set of sampled model parameters $\{\omega_n\}$. Given that the objective function in Equation (9.4) is non-convex, we solve it by finding multiple local optima with random restarts. Let $\boldsymbol{x}^{(t)}$ be the patrolling strategy computed by PAWS-Adapt for round t. The rangers then update their strategy in the new round. As shown in Algorithm 9, the rangers will update the poachers' behavioral models each round after more data is collected. They then switch to a new strategy that was computed with the new model.

9.4 Evaluation

9.4.1 General Game Settings

In the first set of our experiments, we generate a random payoff matrix similar to the experiment in Chapter 5. The crime data points are simulated as the following: given the true distribution of ω, we first draw a set of random parameters for ω to represent the whole poacher population. Let N_p be the total number of poachers. In each round, we first draw a subset of ω from these N_p values to represent the subset of poachers who are going to commit crimes in the current round. Given the patrolling strategy, we then simulated the target choices made by this subset of poachers. These choices are recorded as the anonymous data. Meanwhile, we randomly select a given number of poachers from this subset to represent the poachers that are captured by the rangers in the current round. Once a poacher is captured, the choices he made in the previous round will be linked and recorded as the identified data points.



(a) Cumulative EU

(b) Strategy Convergence

Figure 9.3: Simulation results over round

In Figure 9.3(a), we show the cumulative expected utility (EU) of the rangers over the round. We compare three different approaches: PAWS-Learn, learning from only the identified data, and learning from only the anonymous data. We also included the maximin strategy as the baseline.

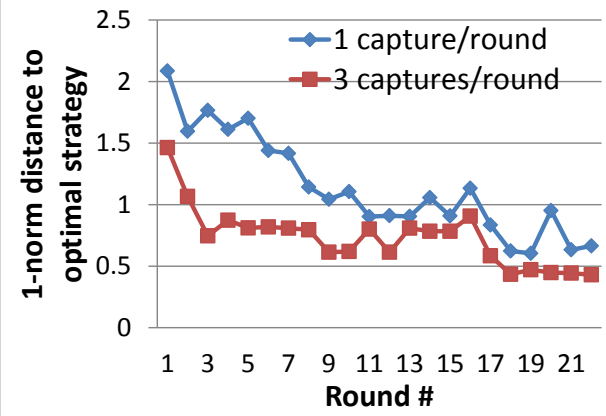


Figure 9.4: Slow Capture v.s. Fast Capture

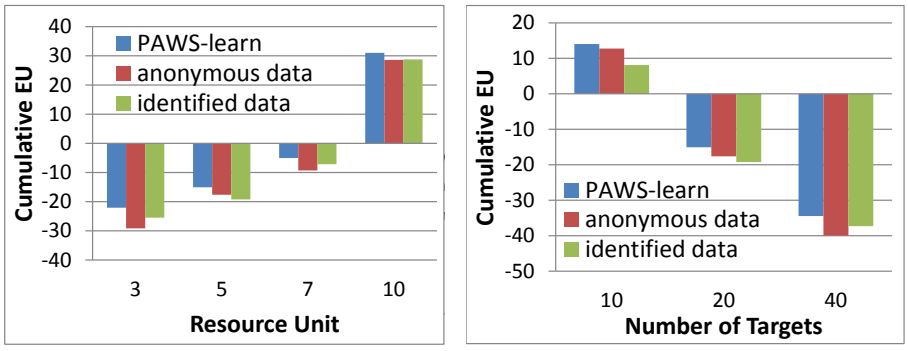
The upper bound is computed assuming the rangers know the true distribution of ω. In Figure 9.3(a), we show the average result over 20 random game instances. We set the number of targets to 20 and the number of security resources to 5. The true distribution of ω is the same as that learned in Section 9.2.2. In each round, 50 anonymous data points are generated, and 3 poachers are captured. As can be seen in the figure, PAWS-Learn outperforms the other two learning approaches that use one type of data. Furthermore, learning indeed helps improve the patrolling strategy since the three solid lines are much closer to the upper bound compared to the baseline solution maximin strategy.

In Figure 9.3(b), we show the convergence of the patrolling strategy from the three different learning methods. The figure shows that PAWS-Learn converges faster than the other two methods. Thus, combining the two types of data indeed boosts the learning of the poacher population's behavioral model.

In order to show how the speed of capturing poachers impacts the performance of PAWS, we fix the number of anonymous data points to 50 and simulate the captured poachers in each round at two different paces: 1 poacher vs. 3 poachers. Figure 9.4 shows the convergence of

PAWS-Learn in these two cases. It is clear that the strategy converges faster if more poachers are captured.

We compare the cumulative EU achieved by the three different methods under varying number of targets and varying amount of resources. In both Figure 9.5(a) and 9.5(b), the y-axis displays the cumulative EU of the rangers at the end of round 20. In both figures, we simulate 50 crimes and randomly generate 3 captured poachers each round. In Figure 9.5(a), we vary the number of resources on the x-axis while fixing the number of targets to 20. It shows that the cumulative EU increases as more resources are added. In addition, PAWS-Learn outperforms the other two approaches regardless of resource quantity. Similarly, we vary the number of targets on the x-axis in Figure 9.5(b) while fixing the amount of resources to 5. The better performance of PAWS-Learn over the other two learning methods can be seen from the figure regardless of the number of targets.



(a) 20 Targets, 3 Captures, 50 Crimes (b) 5 security resources, 3 Captures, 50 Crimes

Figure 9.5: Comparing cumulative EU at round 20

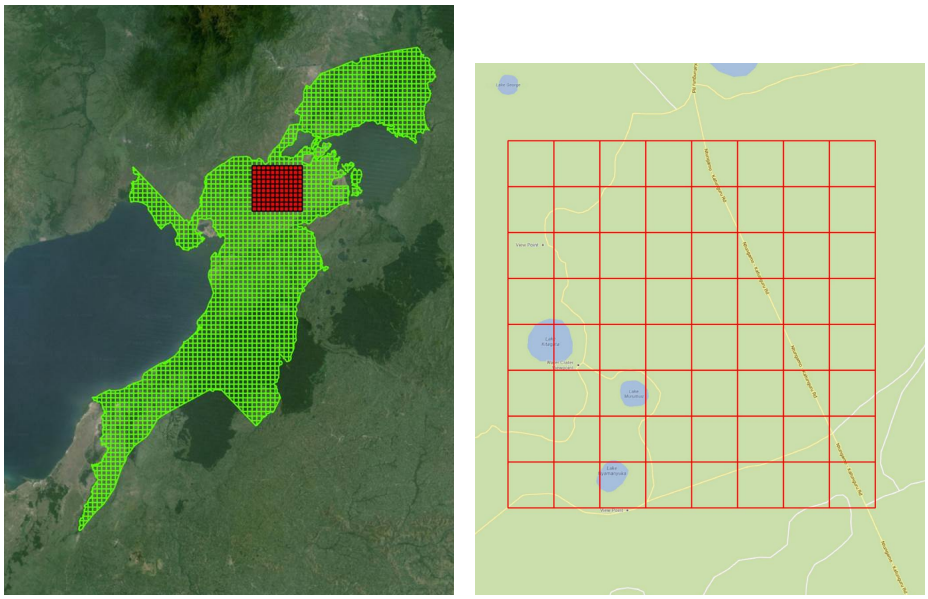
9.4.2 Results for the Deployment Area

We now show the experiment results of applying PAWS to QENP. We focus on a 64 square kilometer area in QENP that features flat grasslands, an international trade route that connects

nearby Democratic Republic of the Congo, smaller roads, and fresh water. In our simulation area 9.6(b), the series of lakes are modeled as areas of high animal density. Since the roads in this area provide multiple access points for poachers and rangers, they can leave the closest road at the closest point to their targeted cells. We calculate travel distances according to that rationale. These representations of animal density and distance form the primary basis for the payoffs for both the rangers and the poachers. The poachers' reward (i.e., if they choose a cell not covered by rangers) depends on the relative animal density of the cell and the travelling cost to that cell. The travelling cost depends on the distance from the closest entry point (e.g., a road). Therefore, a lake close to a road is at high risk for poaching and is thus modelled as an area of high reward to the poachers. In turn, the poachers' penalty (i.e., the chosen cell is covered by rangers) is decided by the travelling cost to a cell and the loss of being captured by the rangers. The rangers' reward is considered to be uniform since their goal is to search for snares and capture poachers regardless of the location. The penalty for the rangers (i.e., fail to find snares at a place) is decided by the animal density of the cell. Further discussion of the rationale can be found in the domain section 9.1.

We run simulations with a sample game, similar to that in the general setting as explained in Section 9.4.1. Figure 9.7 displays the simulation results, where the number of resources is set to 16, indicating that a single patrol covers 16 grid areas in the map. The ranger's cumulative EU is shown in Figure 9.7(a). It can be seen that PAWS-Learn achieves very close performance to the optimal strategy. The convergence of the patrolling strategy to the optimal strategy is shown in Figure 9.7(b).

In order to help visualize the change of ranger's patrolling strategy, we show the coverage density in the 8-by-8 area at three different rounds in Figure 9.8. Darker colors indicate less

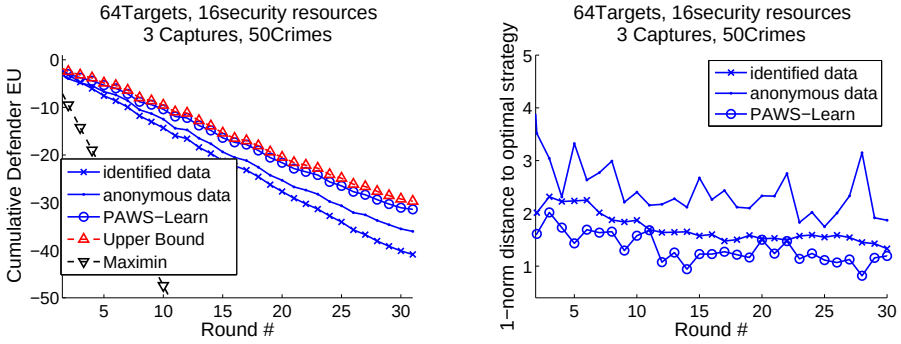


(a) A zoomed out view of the simulation area. (b) The 64 sq. km grid overlayed on the simulation area.

Figure 9.6: The QENP area of interest for our simulation

coverage in the area. Note that there are three lakes located in the lower-left area, where the density of animals is higher. It is clear that these areas are covered more by the rangers. The figure also shows a clear shift of the patrolling coverage over the rounds.

These results are enthusiastically received by our collaborator at QENP. While the existing framework requires manual analysis of the snare data, PAWS provides a systematic way of generating patrolling strategies based on automatic analysis of the data. PAWS will start to be tested in the field in March 2014 with actual deployment planned for the latter portion of 2014.



(a) Cumulative EU (b) Strategy Convergence

Figure 9.7: Simulation results over round for the 64 sq. km grid area in QENP

Round 1 Round 5 Round 20

Figure 9.8: Patrolling coverage density in the park

Chapter 10: Conclusion

Game-theoretic approach has become a very important tool for solving real-world security problems. Its usefulness is proved by a number of real-world deployed applications, including ARMOR [Pita et al., 2008] for the Los Angeles International Airport, IRIS [Tsai et al., 2009] for the Federal Air Marshals, GUARDS [Pita et al., 2011] for the Transportation Security Administrative. These systems have often adopted the standard assumption of a perfectly rational adversary made by the classic game theory, which make not hold in the real-world against human adversaries who may have bounded rationality. While such assumption is a reasonable start for these first generation of applications, it is critical to address the decision making of human adversary as the next step.

My thesis aims to address this challenge by closing the gap between two important subfileds in game theory: Behavioral Game Theory and Algorithmic Game Theory. While the former provides empirical models for predicting the decision making of human players; the latter focuses developing efficient computation of the optimal strategy for the players. In addressing the decision-making of human adversary for real-world security problems, the key is to bridging the efforts from both sides.

To that end, my thesis on the one hand provide novel models for predicting adversary decision making in security games by first investigating the effectiveness of different existing models and then further extending the selected model with key insights draw from data. On the other hand, my thesis develops efficient algorithms for optimizing the defender's resource allocation strategy incorporating the behavioral model of the adversary. In particular, my thesis has the following five key contributions.

10.1 Contributions

- Stochastic model for adversary decision making: This work answers a critical question of which existing model to use to predict adversary decision making. In particular, it presents: (i) new efficient algorithms for computing optimal strategic solutions using Prospect Theory and Quantal Response Equilibrium; (ii) the most comprehensive experiment to date studying the effectiveness of different models against human subjects for security games; and (iii) new techniques for generating representative payoff structures for behavioral experiments in generic classes of games. Our results with human subjects show that our new techniques outperform the leading contender for modeling human behavior in security games.

- More sophisticated models: This work analyzes the effectiveness of adversary model in security games towards addressing the the bounded rationality of adversary. Through extensive experiments with human subjects, I emphatically answer the question in the affirmative, while providing the following key results: (i) our algorithm, SU-BRQR, based on

a novel integration of human behavior model with the subjective utility function, significantly outperforms an robust optimization approach MATCH; (ii) we are the first to present experimental results with security intelligence experts, and find that even though the experts are more rational than the Amazon Turk workers, SU-BRQR still outperforms an approach assuming perfect rationality (and to a more limited extent MATCH); (iii) we show the advantage of SU-BRQR in a new, large game setting and demonstrate that sufficient data enables it to improve its performance over MATCH.

- GOSAQ and PASAQ: I provide two algorithms for efficient computation of the defenders' optimal strategy incorporating a boundedly rational model of the adversary. They overcomes the difficulties of solving a nonlinear and non-convex optimization problem and handling constraints on assigning security resources in designing defender strategies. In addressing these difficulties, GOSAQ guarantees the global optimal solution in computing the defender strategy against an adversarys quantal response; PASAQ provides more efficient computation of the defender strategy with nearly-optimal solution quality. Both algorithms achieves much better solution quality than the benchmark algorithm BRQR. In the presence of resource assignment constraint PASAQ is shown to achieves much better computational efficiency than both GOSAQ and a benchmark algorithm BRQR. In fact, the approximation error of PASAQ is proven to be linearly bounded by the piecewise linear accuracy.

- BLADE: I develop the algorithm to further scale-up for computing defender optimal strategy in massive security games with trillions of defender strategies incorporating the bounded

rationality of the adversary. BLADE is based on three novel ideas. First, we present a separation oracle that can effectively prune the search space via deep cuts. More importantly we show that to handle massive scale SSGs, not only must this separation oracle itself use a secondary oracle but that this two-level hierarchy of oracles is efficient. Second, we provide a novel heuristic to further speed-up BLADE by exploiting the SSG objective function to improve its cuts. Third, BLADE provides a technique for quality-efficiency tradeoff. As we experimentally demonstrate, BLADE is significantly more efficient than a Branch-and-Price based algorithm.

- Adaptive resource allocation: In domains with collective data of adversary events, we are facing new challenges including learning the behavioral model of adversary from the collected data. Using wildlife protection as an example domain, I present PAWS, a novel application for improving wildlife crime patrols, which is essential to combating wildlife poaching. As demonstrated in the experimental results, PAWS successfully models the wildlife crime domain and optimizes wildlife crime patrols while remaining flexible enough to operate generally and in a specific deployed area. Due to the unique challenges introduced by wildlife crime, we have also made a series of necessary technical contributions. Specifically, the success of PAWS depend on the following novel contributions: 1. a stochastic behavioral model extension that captures the populations heterogeneity; 2. PAWS-Learn, which combines both anonymous and identified data to improve the accuracy of the estimated behavioral model; 3. PAWS-Adapt, which adapts the rangers patrolling strategy against the behavioral model generated by PAWS-Learn.

10.2 Future Work

In this thesis, I have shown how game-theoretic approach can be applied for optimizing resource allocation in security problems, with a focus on two domains: counter-terrorism and preventing illegal poaching of wildlife. As security remains an important global concern, there are numerous research opportunities available.

One key area is to translate the results obtained here in controlled experiments on AMT into specific, real-world security applications. Most of the issues related to making this transition are not unique to our work, but apply more generally to studies in agent/human interactions. For example, the specific conditions tested in the lab and the way in which decisions are presented is not likely to be exactly reflected in real interactions, and neither is the population of adversaries identical to the population of adversaries in a real-world security setting. However, our methods are based on fundamental features of human decision-making that are robustly supported in a large number of behavioral studies and these methods would thus translate into real-world applications. In addition, the parameters offer some ability to tune the models over time to specific settings or populations of interest, and our methodology provides techniques for tuning these parameters. The parameter settings in our work can serve as initial settings in a real deployment to be adapted over time. Alternatively, the parameters can initially be set conservatively (e.g., somewhat close to settings that result in a standard equilibrium), and adapted over time from this starting point. Another interesting possibility that could be explored in future work is to develop ways to incorporate different sources of information (such as prior knowledge of the biases of specific adversaries) into the models in a general way.

One other possible direction for future work concerns with further improvements of the model in PAWS. The current model in PAWS is based on a set of simplifying assumptions about the domain. For example, the adversaries is now assumed to be myopic and have a static behavioral model. However, in reality, as the defender gradually improves the behavioral model of the adversaries, the adversaries might also adapt their strategies. Considering such dynamics in PAWS and other security games with repeated settings will be critical to improve the performance of the model. In addition, PAWS doesn't consider the collaboration and competition among the individual poachers. Instead, it is assumed that individual poachers make independent decisions during the illegal poaching. Modeling the collaboration and competition among individual poachers will be a necessary next step to improve the performance of PAWS. Furthermore, the current learning model assumes that the rangers have perfect knowledge of the location distribution of the past poaching events. However, given that the rangers only find proof of poaching events in areas where they go for patrolling, the distribution of the poaching events in the uncovered area is in fact unknown to the rangers. A necessary next step is to modify the learning model to take into consideration such noise in the data.

Another possible direction relates to the efficient computation of defender optimal strategy. My algorithms open the door for optimizing defender resource allocation in massive real-world security problems with large number of defender pure strategies. With problems similar to TRUSTS [Yin et al., 2012] with large number of adversary pure strategies, the door remains open for specialized techniques to further improve the computational efficiency of the algorithms. Furthermore, BLADE assumes a single type of adversary behavioral model. In domains with a large population of adversaries such as wildlife protection, efficient algorithm is still need to address the Bayesian types of adversary behavioral model.

In the long run, a general framework should be built for applying game-theoretic approach to any security domains incorporating the behavioral model of human decision-making. First, a behavioral model will need to be developed for predicting adversary behavioral accounting for the new features that might be involved in the decision-making process of the adversaries in these new domains. One possible approach is to use a quantitative model similar to the quantal response model. The many features that might be involved in the decision making process of the adversary can be built into the model using machine learning methods if data is available. Once a model is developed for predicting the adversary behavior, efficient algorithms for optimizing defender resource allocation will need to be developed. Similar approaches applied in this thesis for developing PASAQ and BLADE can potentially be applied given that the quantitative model of adversary decision making usually leads to a non-convex optimization problem.

Bibliography

Max Abrahms. What terrorists really want: Terrorist motives and counterterrorism strategy. *International Security*, 32(4):78–105, 2008.

Michele Aghassi and Dimitris Bertsimas. Robust game theory. *Math. Program.*, 107:231–273, 2006.

Noa Agmon, Sarit Kraus, and Gal A. Kaminka. Multi-robot perimeter patrol in adversarial settings. In *In ICAT*, 2008.

Noa Agmon, Sarit Kraus, Gal A. Kaminka, and Vladimir Sadow. Adversarial uncertainty in multi-robot patrol. In *In IJCAI*, 2009.

S. A. Ali. Rs 18L seized in nakabandi at Vile Parle. *Times of India*, August 2009.

Graham Allison and Philip Zelikow. *Essence of Decision: Explaining the Cuban Missile Crisis*. Pearson, 1999.

Bo An, Manish Jain, Milind Tambe, and Christopher Kiekintveld. Mixed-initiative optimization in security games: A preliminary report. *In Proceeding of the AAAI Spring Symposium*, 2010.

Bo An, David Kempe, Christopher Kiekintveld, Eric Shieh, Satinder Singh, Milind Tambe, and Yevgeniy Vorobeychik. Security games with limited surveillance. In *In AAAI*, 2012.

Robert J. Aumann and M. B. Maschler. *Repeated Games with Incomplete Information*. The MIT press, 1995.

Amos Azaria, Zinovi Rabinovich, Sarit Kraus, and Claudia V. Goldman. Strategic information disclosure to people with multiple alternatives. In *In AAAI*, pages 594–600, 2011.

Amos Azaria, Zinovi Rabinovich, Sarit Kraus, and Claudia V. Goldman. Strategic information disclosure to people with multiple alternatives. In *In AAAI*, 2012.

C. Barnhart, E. Johnson, g. Nemhauser, M. Savelsbergh, and P. Vance. Branch and price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1994.

Nicola Basiloco, Nicola Gatti, and Francesco Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *In AAMAS*, 2009.

J. C. Becsey, Laszlo Berke, and James R. Callan. Nonlinear least squares methods: A direct grid search approach. *Journal of Chemical Education*, 45(11):728, 1968.

S. Boyd and L. Vandenberghe. Localization and cutting-plane methods. *Lecture Notes*, 2008.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, 2004.

M. Breton, A. Alg, and A. Haurie. Sequential stackelberg equilibria in two-person games. *Optimization Theory and Applications*, 59(1):71–97, 1988.

George W. Brown. Iterative solution of games by fictitious play. In *Activity Analysis of Production and Allocation*. Wiley, 1951.

R. G. Burgess and C. J. Darken. Realistic human path planning using fluid simulation. In *Proceedings of Behavior Representation in Modeling and Simulation (BRIMS)*, 2004.

Richard H. Byrd, Jorge Nocedal, and Richard A. Waltz. Knitro: An integrated package for nonlinear optimization. In *Large-Scale Nonlinear Optimization*, pages 35–59. G. di Pillo and M. Roma, eds, Springer-Verlag, 2006.

Colin F. Camerer. *Behavioral Game Theory: Experiments in Strategic Interaction*. Princeton University Press, Princeton, New Jersey, 2003.

Colin F. Camerer, Teck-Hua Ho, and Juin-Kuan Chongn. A congnitive hierarchy model of games. *QJE*, 119(3):861–898, 2004.

R. Chandran and G. Beitchman. Battlefor mumbai ends, death toll rises to 195. *Times of India*, November 2008.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, Cambridge, Massachusetts, USA, 2006.

Vincent Conitzer and Thomas Sandholm. Computing the optimal strategy to commit to. In *In Proceedings of the ACM Conference on Electronic Commerce (ACM-EC)*, pages 82–90, 2006.

Miguel Costa-Gomes, Vicent P. Crawford, and Brimp Broseta. Cognition and behavior in normal-form games: An experimental studys. *Econometrica*, 69(5):1193–1235, 2001.

Celso de Melo, Peter Carnevale, and Jonathan Gratch. The effect of expression of anger and happiness in computer agents on negotiations with humans. In *In AAMAS*, 2011.

Enrico Diecidue and Peter P. Wakker. On the intuition of rank-dependent utility. *The Journal of Risk and Uncertainty*, 23(3):281–289, 2001.

Nick Feltovich. Reinforment-based vs. belief-based learning models in experimental asymmetric-information games. *Econometrica*, 68(3):605–641, May 2000.

Sevan G. Ficici and Avi Pfeffer. Simultaneously modeling humans' preferences and their beliefs about others' preferences. In *In AAMAS*, 2008.

Baruch Fischhoff, Bernard Goitein, and Zur Shapira. Subjective expected utility: A model of decision-making. *Journal of American Society of Information Science*, 32(5):391–399, 1981.

Nicola Gatti. Game theoretical insights in strategic patrolling: Model and algorithm in normal-form. In *In ECAI-08*, pages 403–407, 2008a.

Nicola Gatti. Game theoretical insights in strategic patrolling model and algorithm in normal-form. In *In ECAI*, pages 403–407, 2008b.

G. Gigerenzer, Todd P. M., and the ABC Research Group. *Simple Heuristics that make us smart*. Oxford University Press, 1999.

Paul Gill and Joseph Young. Comparing role-specific terrorist profiles. In *American Society of Criminology Annual Meeting*, 2011.

Jacob K. Goeree, Charles A. Holt, and Thomas R. Palfrey. Regular quantal response equilibrium. *Experimental Economics*, 8(4):347–367, December 2005.

Priscilla E. Greenwood and Michael S. Nikulin. *A Guide to Chi-squared Testing*. John Wiley & Sons, Inc, 1996.

Reid Hastie and Robyn M. Dawes. *Rational Choice in an Uncertain World: the Psychology of Judgement and Decision Making*. Sage Publications, Thounds Oaks, 2001.

Manish Jain, Erim Kardes, Christopher Kiekintveld, Milind Tambe, and Fernando Ordonez. Security games with arbitrary schedules: A branch and price approach. In *In AAAI*, 2010a.

Manish Jain, James Pita, Jason Tsai, Christopher Kiekintveld, Shyamsunder Rathi, Fernando Ordóñez, and Milind Tambe. Software assistants for patrol planning at lax and federal air marshals service. *Interfaces*, 40(4):267–290, 2010b.

Manish Jain, Dmytro Korzhyk, Ondrej Vanek, Vincent Conitzer, Michal Pechoucek, and Milind Tambe. A double oracle algorithm for zero-sum security games on graphs. *In AAMAS*, 2011a.

Manish Jain, Milind Tambe, and Christopher Kiekintveld. Quality-bounded solutions for finite bayesian stackelberg games: Scaling up. In *In AAMAS*, 2011b.

Manish Jain, Kevin Leyton-Brown, and Milind Tambe. The deployment-to-saturation ratio in security games. *In AAAI*, 2012.

Daniel Kahneman and Amos Tvesky. Prospect theory: An analysis of decision under risk. *Econometrica*, 47(2):263–292, 1979.

Daniel Kahneman and Amos Tvesky. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and Uncertainty*, 5:297–322, 1992.

Gregory L. Keeney and Detlof von Winterfeldt. Identifying and structuring the objectives of terrorists. *Risk Analysis*, 30(12):1803–1816, 2010.

J. E. Kelley. The cutting-plane method for solving convex programs. *The Society for Industrial and Applied Mathematics*, 8(4):703–713, 1960.

Christopher Kiekintveld, Manish Jain, Jason Tsai, James Pita, Fernando Ordóñez, and Milind Tambe. Computing optimal randomized resource allocations for massive security games. In *In AAMAS*, pages 689–696, 2009.

Christopher Kiekintveld, Janusz Marecki, and Milind Tambe. Approximation methods for infinite bayesian stackelberg games: Modeling distributional payoff uncertainty. In *In AAMAS*, 2011.

Dmytro Korzhyk, Vincent Conitzer, and Ronald Parr. Complexity of computing optimal stackelberg strategies in security resource allocation games. In *In AAAI*, 2010.

Joshua Letchford, Vincent Conitzer, and Kamesh Munagala. Learning and approximating the optimal strategy to commit to. In *In Algorithmic Game Theory*. Springer, 2009.

Janusz Marecki, Gerry Tesauro, and Richard Segal. Playing repeated stackelberg games with unknown opponents. In *In AAMAS*, 2012.

Winter Mason and Siddharth Suri. Conducting behavioral research on amazons mechanical turk. *Behavior Research Methods*, 44(1):1–23, 2012.

Daniel L. McFadden. Econometric analysis of qualitative response models. *Handbook of Econometrics*, 2:1395–1457, 1984.

Daniel L. McFadden. A method of simulated moments for estimation of discrete choice models without numerical integration. *Econometrica*, 57(5):995–1026, 1989.

Richard D. McKelvey and Thomas R. Palfrey. Quantal response equilibria for normal form games. *Games and Economic Behavior*, 2:6–38, 1995.

Moses Montesh. Rhino poaching: A new form of organised crime. Technical report, College of Law Research and Innovation Committee of the University of South Africa, 2013.

William Moreto. *To Conserve and Protect: Examining Law Enforcement Ranger Culture and Operations in Queen Elizabeth National Park, Uganda*. Thesis, Rutgers, 2013.

Thanh H. Nguyen, Rong Yang, Amos Azaria, Sarit Kraus, and Milind Tambe. Analyzing the effectiveness of adversary modeling in security games. In *In AAAI*, 2013.

E. Nudelman, J. Wortman, Y. Shoham, and K. Leyton-Brown. Run the gamut: A comprehensive approach to evaluating game-theoretic algorithms. In *AAMAS*, pages 880–887, 2004.

Christos H. Papadimitriou and Tim Roughgarden. Computing correlated equilibria in multi-player games. *Journal of the ACM*, 55(3):14, July 2008.

Praveen Paruchuri, Jonathan P. Pearce, Janusz Marecki, Milind Tambe, Fernando Ordóñez, and Sarit Kraus. Playing games for security: An efficient exact algorithm for solving bayesian stackelberg games. In *In AAMAS*, 2008.

Noam Peled, Ya'akov Gal, and Sarit Kraus. A study of computational and human strategies in revelation games. In *In AAMAS*, 2011.

James Pita, Manish Jain, Fernando Ordóñez, Christopher Portway, Milind Tambe, Craig Western, Praveen Paruchuri, and Sarit Kraus. Deployed armor protection: The application of a game theoretic model for security at the los angeles international airport. In *In AAMAS*, 2008.

James Pita, Manish Jain, Fernando Ordóñez, Milind Tambe, and Sarit Kraus. Solving stackelberg games in the real-world: Addressing bounded rationality and limited observations in human preference models. *Artificial Intelligence Journal*, 174(15):1142–1171, 2010.

James Pita, Milind Tambe, Chris Kiekintveld, Shane Cullen, and Erin Steigerwald. Guards - game theoretic security allocation on a national scale. In *In AAMAS*, 2011.

James Pita, Richard John, Rajiv Maheswaran, Milind Tambe, and Sarit Kraus. A robust approach to addressing human adversaries in security games. In *ECAI*, pages 660–665, 2012.

Mark R. Pogrebin. *About Criminals: A View of the Offenders World*. SAGE, 2012.

Yundi Qian, William B. Haskell, Albert Xin Jiang, and Milind Tambe. Online planning for optimal protector strategies in resource conservation games. In *In AAMAS*, 2014.

Ulf-Dietrich Reips. Conducting behavioral research on amazons mechanical turk. *Experimental Psychology*, 49(4):243–256, 2002.

Louise Richardson. *What Terrorists Want: Understanding the Enemy, Containing the Threat*. Random House Trade Paperbacks, 2007.

Heather Rosoff and Richard John. Decision analysis by proxy for the rational terrorist. In *In QRASA at IJCAI*, pages 25–32, 2009.

Ariel Rubinstein. *Modeling Bounded Rationality*. MIT Press, Cambridge, Massachusetts, 1998.

P. S. Sastry, V. V. Phansalkar, and M. Thathachar. Decentralized learning of nash equilibria in multi-person stochastic games with incomplete information. *IEEE TRANSACTIONS ON SYSTEMS MAN AND CYBERNETICS*, 24(5), 1994.

Leonard J. Savage. *The Foundations of Statistics*. Dover Publications, 1972.

Mrinal K. Sen and Paul L. Stoffa. *Global optimization methods in geophysical inversion*. Elsevier, New York, 1995.

Eric Shieh, Bo An, Rong Yang, Milind Tambe, Craig Baldwin, Joseph DiRenzo, Ben Maule, and Garrett Meyer. Protect: A deployed game theoretic system to protect the ports of the united states. In *In AAMAS*, 2012.

Herbert A. Simon. Rational choice and the structure of the environment. *Psychological Review*, 63(2):129–138, 1956.

Herbert A. Simon. *Science of the Artificial*. MIT Press, Cambridge, Massachusetts, 1969.

Dale O. Stahl and Paul W. Wilson. Experimental evidence on players' models of other players. *JEBO*, 25(3):309–327, 1994.

Chris Starmer. Developments in non-expected utility theory: The hunt for a descriptive theory of choice under risk. *Journal of Economic Literature*, 38(2):332–382, 2000.

Donald Stevens, Thomas Hamilton, Marvin Schaffer, Diana Dunham-Scott, Jamison J. Medby, Edward W. Chan, John Gibson, Mel Eisman, Richard Mesic, Charles T. Kelly Jr, Julie Kim, Tom LaTourrette, and J. Jack Riley. *Implementing security improvement options at Los Angeles International Airport*. RAND Corporation, 2006. URL `http://www.rand.org/pubs/documentedbriefings/2006/RANDDB499-1.pdf`.

Kenneth Train. *Discrete Choice Methods with Simulation*. Cambridge University Press, Cambridge, UK, 2003.

Jason Tsai, Shyamsunder Rathi, Christopher Kiekintveld, Fernando Ordóñez, and Milind Tambe. Iris - a tool for strategic security allocation in transportation networks. In *In AAMAS*, 2009.

Jason Tsai, Zhengyu Yin, Jun young Kwak, David Kempe, Christopher Kiekintveld, and Milind Tambe. Urban security: Game-theoretic resource allocation in networked physical domains. In *In AAAI*, 2010.

Stephen A. Vavasis. Complexity issues in global optimization: a survey. In *Handbook of Global Optimization*, pages 27–41. In R. Horst and P.M. Pardalos, editors, Kluwer, 1995.

Heinrich Freiherr von Stackelberg. *Market Structure and Equilibrium*. Springer, 2011.

Bernhard von Stengel and Shmuel Zamir. Leadership with commitment to mixed strategies. In *Tech. rep. LSE-CDAM-2004-01, CDAM Research Report*, 2004.

Alan Washburn and Kevin Wood. Two-person zero-sum games for network interdiction. *Operations Research*, 43(2):243–251, 1995.

Yussuf Adan Wato, Geoffrey M. Wahungu, and Moses Makonjio Okello. Correlates of wildlife snaring patterns in tsavo west national park, kenya. *Biological Conservation*, 132(4):500–509, 2006. ISSN 0006-3207. doi: http://dx.doi.org/10.1016/j.biocon.2006.05.010. URL `http://www.sciencedirect.com/science/article/pii/S0006320706002047`.

Rand R. Wilcox. *Applying contemporary statistical techniques*. Academic Press, 2003.

James R. Wright and Kevin Leyton-Brown. Beyond equilibrium: Predicting human behavior in normal-form games. In *In AAAI*, 2010.

Rong Yang, Christopher Kiekintveld, Fernando Ordóñez, Milind Tambe, and Richard John. Improving resource allocation strategy against human adversaries in security games. In *In IJCAI*, 2011.

Rong Yang, Albert Xin Jiang, Fei Fang, Milind Tambe, Rajiv Maheswaran, and Karthik Rajagopal. Designing better strategies against human adversaries in network security games. In *In AAMAS*, 2012a.

Rong Yang, Fernando Ordóñez, and Milind Tambe. Computing optimal strategy against quantal response in security games. In *In AAMAS*, 2012b.

Rong Yang, Albert Xin Jiang, Milind Tambe, and Fernando Ordonez. Scaling-up security games with boundedly rational adversaries: A cutting-plane approach. In *In IJCAI*, 2013a.

Rong Yang, Christopher Kiekintveld, Fernando Ordóñez, Milind Tambe, and Richard John. Improving resource allocation strategy against human adversaries in security games: An extended study. *AIJ*, 195:440–469, February 2013b.

Rong Yang, Benjamin Ford, Milind Tambe, and Andrew Lemieux. Adaptive resource allocation for wildlife protection against illegal poachers. In *In AAMAS*, 2014.

Zhengyu Yin and Milind Tambe. A unified method for handling discrete and continuous uncertainty in bayesian stackelberg games. In *In AAMAS*, 2012.

Zhengyu Yin, Dmytro Korzhyk, Christopher Kiekintveld, Vincent Conitzer, and Milind Tambe. Stackelberg vs. nash in security games: Interchangeability, equivalence, and uniqueness. In *In AAMAS*, 2010.

Zhengyu Yin, Manish Jain, Milind Tambe, and Fernando Ordóñez. Risk-averse strategies for security games with execution and observational uncertainty. In *In AAAI*, 2011.

Zhengyu Yin, Albert Jiang, Matthew Johnson, Milind Tambe, Christopher Kiekintveld, Kevin Leyton-Brown, Tuomas Sandholm, and John Sullivan. Trusts: Scheduling randomized patrols for fare inspection in transit systems. In *In IAAI*, 2012.

Appendix A: Error Bound of PASAQ

For simplicity, let's first define the following notations:

- $F^{(r)}(x)$, the objective function of the `CF-OPT` problem associated with a given estimation value r:
$$F^{(r)}(x) = \sum_{i \in \mathcal{T}} \theta_i (r - P_i^d) e^{-\beta_i x_i} - \sum_{i \in \mathcal{T}} \theta_i \alpha_i x_i e^{-\beta_i x_i}$$
- $\nu^{(r)} = arg\min_x F^{(r)}(x)$
- $\tilde{F}^{(r)}(x)$, the objective function of the `PASAQ-MILP` problem associated with a given estimation value r:
$$\tilde{F}^{(r)}(x) = \sum_{i \in \mathcal{T}} \theta_i (r - P_i^d)(1 + \sum_{k=1}^{K} a_{il} x_{il}) - \sum_{i \in \mathcal{T}} \theta_i \alpha_i \sum_{k=1}^{K} b_{il} x_{il}$$
- $\tilde{\nu}^{(r)} = arg\min_x \tilde{F}^{(r)}(x)$

Also, we define the game constants decided by the payoff in Table A.1

Lemma 14. *For any real value $r \in \mathscr{R}$, one of the following two conditions holds.*
(a) $r \leq p^* \iff \exists \mathbf{x} \in \mathscr{X}_f$, s.t., $rD(\mathbf{x}) - N(\mathbf{x}) \leq 0$;
(b) $r > p^* \iff \forall \mathbf{x} \in \mathscr{X}_f$, $rD(\mathbf{x}) - N(\mathbf{x}) > 0$

Proof. We only prove (a) as (b) is proven similarly.

'$\Leftarrow$': since $\exists x$ such that $rD(\boldsymbol{x}) \leq N(\boldsymbol{x})$, this means that $r \leq \frac{N(\boldsymbol{x})}{D(\boldsymbol{x})} \leq p^*$;

'$\Rightarrow$': Since `P1` optimizes a continuous objective over a closed convex set, then there exists an optimal solution $\boldsymbol{x}^*$ such that $p^* = \frac{N(\boldsymbol{x}^*)}{D(\boldsymbol{x}^*)} \geq r$ which rearranging gives the result. □

Table A.1: Game Constant

$\overline{\theta} := \max_{i \in \mathcal{T}} \theta_i$	$\underline{\theta} := \min_{i \in \mathcal{T}} \theta_i$
$\overline{R^d} := \max_{i \in \mathcal{T}} \lvert R_i^d \rvert$	$\overline{P^d} := \max_{i \in \mathcal{T}} \lvert P_i^d \rvert$
$\overline{\beta} := \max_{i \in \mathcal{T}} \beta_i$	$\overline{\alpha} := \max_{i \in \mathcal{T}} \alpha_i$

Lemma 15. *The approximation error of the piecewise linear function is bounded as the following:*

$$|e^{-\beta_i x_i} - L_i^{(1)}(x_i)| \leq \frac{\beta_i}{K}, 0 \leq x_i \leq 1, \quad \forall i \in \mathcal{T} \tag{A.1}$$

$$|x_i e^{-\beta_i x_i} - L_i^{(2)}(x_i)| \leq \frac{1}{K}, 0 \leq x_i \leq 1, \quad \forall i \in \mathcal{T} \tag{A.2}$$

Proof. Let $f_i(x_i)$ be the original function, and $L_i(x_i)$ be the corresponding piecewise linear approximation function. The following proof holds for both $f_i(x_i) = e^{-\beta_i x_i}$ and $f_i(x_i) = x_i e^{-\beta_i x_i}$:

$$\max_{0 \leq x_i \leq 1} |f_i(x_i) - L_i(x_i)| = \max_{k=1}^{K} \max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} |f_i(x_i) - L_i(x_i)| \tag{A.3}$$

We now prove the bound on $\max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} |f_i(x_i) - L_i(x_i)|$ in three steps:

1. Assuming $f_i(x_i) \geq L_i(x_i), \frac{k-1}{K} \leq x_i \leq \frac{k}{K}$

$$\begin{aligned}
&\max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} |f_i(x_i) - L_i(x_i)| \leq \max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) - \min_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} L_i(x_i) \\
&= \max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) - \min\{L_i(\frac{k-1}{K}), L_i(\frac{k}{K})\} \\
&= \max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) - \min\{f_i(\frac{k-1}{K}), f_i(\frac{k}{K})\} \\
&\leq \max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) - \min_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) \leq \frac{1}{K} \max_{0 \leq x_i \leq 1} |f_i'(x_i)|
\end{aligned}$$

2. Assuming $f_i(x_i) \leq L_i(x_i), \frac{k-1}{K} \leq x_i \leq \frac{k}{K}$

$$\begin{aligned}
&\max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} |f_i(x_i) - L_i(x_i)| \leq \max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} L_i(x_i) - \min_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) \\
&= \max\{L_i(\frac{k-1}{K}), L_i(\frac{k}{K})\} - \min_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) \\
&= \max\{f_i(\frac{k-1}{K}), f_i(\frac{k}{K})\} - \min_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) \\
&\leq \max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) - \min_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} f_i(x_i) \leq \frac{1}{K} \max_{0 \leq x_i \leq 1} |f_i'(x_i)|
\end{aligned}$$

3. If $f_i(x_i)$ and $L_i(x_i)$ get across in $[\frac{k-1}{K}, \frac{k}{K}]$, we could partition the range into small regions such that within each sub partition, the two functions do not get across. We then can apply (a) or (b) within each partition.

Combining the above three conditions, we have

$$\max_{\frac{k-1}{K} \leq x_i \leq \frac{k}{K}} |f_i(x_i) - L_i(x_i)| \leq \frac{1}{K} \max_{0 \leq x_i \leq 1} |f_i'(x_i)| \tag{A.4}$$

At the same time, it can be shown with some effort that

where, $f_i'(x_i)$ is the first order derivative of function $f_i(x_i)$. Combining with Equation (A.3), we have

$$\max_{0\le x_i\le 1} |f_i(x_i) - L_i(x_i)| \le \frac{1}{K} \max_{0\le x_i\le 1} |f_i'(x_i)| \tag{A.5}$$

Hence, the approximation error bound is decided by the maximum absolute value of the first order derivative. It can be shown that

$$\max_{0\le x_i\le 1} |\frac{d(e^{-\beta_i x_i})}{dx_i}| = |\frac{d(e^{-\beta_i x_i})}{dx_i}|_{x_i=0} = \beta_i \tag{A.6}$$

$$\max_{0\le x_i\le 1} |\frac{d(x_i e^{-\beta_i x_i})}{dx_i}| = |\frac{d(x_i e^{-\beta_i x_i})}{dx_i}|_{x_i=0} = 1 \tag{A.7}$$

Combining Equation (A.5)-(A.7) gives the result. □

Lemma 16. *Let L^* and U^* be the lower and upper bounds of* GOSAQ *when the algorithm stops, and x^* is the defender strategy returned by* GOSAQ*. Then,*

$$L^* \le Obj_{P1}(x^*) \le U^*$$

Proof. When the algorithm stops, we have $F^{(L^*)}(x^*) \le 0 \Rightarrow L^* \le \frac{N(x^*)}{D(x^*)} = Obj_{P1}(x^*)$ At the same time, $F^{(U^*)}(x^*) > 0, \forall x \Rightarrow U^* > \frac{N(x^*)}{D(x^*)} = Obj_{P1}(x^*)$ □

Lemma 17. *Let $\tilde{L}^*$ and $\tilde{U}^*$ be the lower and upper bounds of* PASAQ *when the algorithm stops, and $\tilde{Obj}_{P1}(x)$ be the approximation of objective function `P1` with the piecewise linear representation of $e^{-\beta_i x_i}$ and $x_i e^{-\beta_i x_i}$. Then,*

$$\tilde{L}^* \le \tilde{Obj}_{P1}(\tilde{x}^*) \le \tilde{U}^*$$

where, $\tilde{x}^*$ is the defender strategy returned by PASAQ*.*

Proof. Same as that for Lemma 16 □

Lemma 18. *Let $Obj_{P1}(x)$ be the objective function of `P1` and $\tilde{Obj}_{P1}(x)$ be the corresponding approximation with the piecewise linear representation of $e^{-\beta_i x_i}$ and $x_i e^{-\beta_i x_i}$. Then, $\forall x \in \mathscr{X}_f$*

$$|Obj_{P1}(x) - \tilde{Obj}_{P1}(x)| \le (\overline{\theta}/\underline{\theta}) e^{\overline{\beta}} \overline{\beta} \{\overline{R^d} + \overline{P^d}\} + \frac{\overline{\alpha}}{\overline{\beta}}\} \cdot \frac{1}{K} \tag{A.8}$$

Proof. Let $\tilde{N}(x) = \sum_{i\in\mathcal{T}} \theta_i \alpha_i L_i^{(2)}(x_i) + \sum_{i\in\mathcal{T}} \theta_i P_i^d L_i^{(1)}(x_i)$ and $\tilde{D}(x) = \sum_{i\in\mathcal{T}} \theta_i L_i^{(1)} > 0$ be the piecewise linear approximation of the numerator and denominator of Obj_{P1} respectively.

$$\begin{aligned}
&|Obj_{P1}(x) - \tilde{Obj}_{P1}(x)| = |\frac{N(x)}{D(x)} - \frac{\tilde{N}(x)}{\tilde{D}(x)}| \\
&= |\frac{N(x)}{D(x)} - \frac{N(x)}{\tilde{D}(x)} + \frac{N(x)}{\tilde{D}(x)} - \frac{\tilde{N}(x)}{\tilde{D}(x)}| \\
&\le |\frac{N(x)}{D(x)} \frac{\tilde{D}(x) - D(x)}{\tilde{D}(x)}| + |\frac{N(x) - \tilde{N}(x)}{\tilde{D}(x)}| \\
&= \frac{1}{\tilde{D}(x)} (|Obj_{P1}(x)| \cdot |D(x) - \tilde{D}(x)| + |N(x) - \tilde{N}(x)|)
\end{aligned}$$

Based on Lemma 15,

$$|N(x) - \tilde{N}(x)| \leq \sum_{i \in \mathcal{T}} \theta_i \alpha_i \frac{1}{K} + \sum_{i \in \mathcal{T}} \theta_i |P_i^d| \frac{\beta_i}{K} \leq (\overline{\theta}\overline{\alpha} + \overline{P^d}\overline{\theta\beta}) \frac{|\mathcal{T}|}{K}$$
$$|D(x) - \tilde{D}(x)| \leq \sum_{i \in \mathcal{T}} \theta_i \frac{\beta_i}{K} \leq (\overline{\theta}/\underline{\theta})\beta \frac{|\mathcal{T}|}{K}$$

At the same time, $|Obj_{P1}(x)| \leq \overline{R^d}$ and $\tilde{D}(x) \geq |\mathcal{T}|\underline{\theta}e^{-\overline{\beta}}$. Hence,

$$|Obj_{P1}(x) - \tilde{Obj}_{P1}(x)| \leq (\overline{\theta}/\underline{\theta})e^{\overline{\beta}}\overline{\beta}\{\overline{R^d} + \overline{P^d} + \frac{\overline{\alpha}}{\overline{\beta}}\} \cdot \frac{1}{K}$$

□

Lemma 19. $\forall x \in \mathscr{X}_f$, *the following condition holds*

$$|F^{(r)}(x) - \tilde{F}^{(r)}(x)| \leq (|r| + \overline{P^d})\overline{\theta} \sum_{i \in \mathcal{T}} \frac{\beta_i}{K} + \overline{\alpha}\overline{\theta}\frac{T}{K} \tag{A.9}$$

Proof. Let $L_i^{(1)}(x_i) = 1 + \sum_{k=1}^{K} a_{il}x_{il}$ be the piecewise linear approximations of function $e^{-\beta_i x_i}$, and $L_i^{(2)}(x_i) = \sum_{k=1}^{K} b_{il}x_{il}$ be that of function $x_i e^{-\beta_i x_i}$. We have

$$\begin{aligned}
&|F^{(r)}(x) - \tilde{F}^{(r)}(x)| \\
&\leq |\sum_{i \in \mathcal{T}} \theta_i (r - P_i^d) e^{-\beta_i x_i} - \sum_{i \in \mathcal{T}} \theta_i (r - P_i^d) L_i^{(1)}(x_i)| + |\sum_{i \in \mathcal{T}} \theta_i \alpha_i x_i e^{-\beta_i x_i} - \sum_{i \in \mathcal{T}} \theta_i \alpha_i L_i^{(2)}(x_i)| \\
&\leq \sum_{i \in \mathcal{T}} \theta_i |r - P_i^d| \cdot |e^{-\beta_i x_i} - L_i^{(1)}(x_i)| + \sum_{i \in \mathcal{T}} \theta_i \alpha_i |x_i e^{-\beta_i x_i} - L_i^{(2)}(x_i)| \\
&\leq (|r| + P^d)\overline{\theta} \sum_{i \in \mathcal{T}} |e^{-\beta_i x_i} - L_i^{(1)}(x_i)| + \overline{\alpha}\overline{\theta} \sum_{i \in \mathcal{T}} |x_i e^{-\beta_i x_i} - L_i^{(2)}(x_i)|
\end{aligned} \tag{A.10}$$

Combining Equation (A.1) and (A.2), we have

$$|F^{(r)}(x) - \tilde{F}^{(r)}(x)| \leq (|r| + P^d)\overline{\theta} \sum_{i \in \mathcal{T}} \frac{\beta_i}{K} + \overline{\alpha}\overline{\theta}\frac{|\mathcal{T}|}{K}$$

□

Lemma 20. *Let $\tilde{L}^*$ be the estimated maximum of $Obj_{P1}(x)$ by running* PASAQ*, then*

$$\tilde{F}^{(\tilde{L}^*)}(x) \geq -\epsilon\overline{\theta}|\mathcal{T}|, \quad \forall x \in \mathscr{X}_f \tag{A.11}$$

Proof. Let U^* and L^* be the upper and lower bound when the algorithm stops. According to Line 3 in Algorithm 1, $U^* - L^* \leq \epsilon$. Furthermore, $U^* > \tilde{L}^* \geq L^*$. Therefore we know$\tilde{L}^* + \epsilon \geq U^*$, so the result of `CheckFeasiblity` with given $\tilde{L}^* + \epsilon$ must be `infeasible`. In other words,

$$\tilde{F}^{(\tilde{L}^* + \epsilon)}(x) > 0, \quad \forall x \in \mathscr{X}_f \tag{A.12}$$

On the other hand,

$$\tilde{F}^{(\tilde{L}^*+\epsilon)}(x) - \tilde{F}^{(\tilde{L}^*)}(x) = \epsilon \sum_{i \in \mathcal{T}} \theta_i L_i^{(1)}(x_i) \leq \epsilon \overline{\theta} |\mathcal{T}|, \forall x \in \mathscr{X}_f \tag{A.13}$$

Combining Equation (A.12) and (A.13)

$$\tilde{F}^{(\tilde{L}^*)}(x) \geq \tilde{F}^{(\tilde{L}^*+\epsilon)}(x) - \epsilon \overline{\theta} |\mathcal{T}| \geq -\epsilon \overline{\theta} |\mathcal{T}|$$

□

Lemma 21. *Let L^* be the estimated maximum of* `P1` *by* GOSAQ

$$L^* - \tilde{L}^* \leq (\overline{\theta}/\underline{\theta}) e^{\overline{\beta}} \{\epsilon + \frac{1}{K}((\overline{R^d} + \overline{P^d})\overline{\beta} + \overline{\alpha})\} \tag{A.14}$$

Proof. According to Lemma 14, $F^{(L^*)}(\nu^{(L^*)}) \leq 0$. At the same time,

$$\begin{aligned} F^{(L^*)}(\nu^{(L^*)}) &= F^{(\tilde{L}^*)}(\nu^{(L^*)}) + (L^* - \tilde{L}^*) \sum_{i \in \mathcal{T}} \theta_i e^{-\beta_i \nu_i^{(L^*)}} \\ \Rightarrow (L^* - \tilde{L}^*) \sum_{i \in \mathcal{T}} \theta_i e^{-\beta_i \nu_i^{(L^*)}} &\leq -F^{(\tilde{L}^*)}(\nu^{(L^*)}) \end{aligned} \tag{A.15}$$

Furthermore, Lemma 19 indicates that

$$\begin{aligned} &- F^{(\tilde{L}^*)}(\nu^{(L^*)}) \leq -\tilde{F}^{(\tilde{L}^*)}(\nu^{(L^*)}) + (|\tilde{L}^*| + \overline{P^d})\overline{\theta} \sum_{i \in \mathcal{T}} \frac{\beta_i}{K} + +\overline{\alpha}\overline{\theta}\frac{|\mathcal{T}|}{K} \\ &\leq -\tilde{F}^{(\tilde{L}^*)}(\nu^{(L^*)}) + \overline{\theta}((\overline{R^d} + \overline{P^d}) \sum_{i \in \mathcal{T}} \frac{\beta_i}{K} + \frac{|\mathcal{T}|}{K}\overline{\alpha}) \\ &\leq -\tilde{F}^{(\tilde{L}^*)}(\nu^{(L^*)}) + \overline{\theta}(\frac{(\overline{R^d} + \overline{P^d})\overline{\beta}}{K} + \frac{\overline{\alpha}}{K})|\mathcal{T}| \end{aligned} \tag{A.16}$$

since $|\tilde{L}^*| \leq \overline{|R_i^d|}$. Combining Equation (A.11),(A.15) and (A.16)

$$(L^* - \tilde{L}^*) \sum_{i \in \mathcal{T}} \theta_i e^{-\beta_i \nu_i^{(L^*)}} \leq \epsilon \overline{\theta} |\mathcal{T}| + \overline{\theta}(\frac{(\overline{R^d} + \overline{P^d})\overline{\beta}}{K} + \frac{\overline{\alpha}}{K})|\mathcal{T}|$$

Furthermore, $\sum_{i \in \mathcal{T}} \theta_i e^{-\beta_i \nu_i^{(L^*)}} \geq T\underline{\theta} e^{-\overline{\beta}}$, so

$$L^* - \tilde{L}^* \leq (\overline{\theta}/\underline{\theta}) e^{\overline{\beta}} \{\epsilon + \frac{1}{K}((\overline{R^d} + \overline{P^d})\overline{\beta} + \overline{\alpha})\}$$

□