Robust Incident Prediction, Resource Allocation & Dynamic Dispatch

By

Ayan Mukhopadhyay

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Computer Science

August 9, 2019

Nashville, Tennessee

Approved:

Yevgeniy Vorobeychik, Ph.D

Gautam Biswas, Ph.D

Abhishek Dubey, Ph.D

Maithilee Kunda, Ph.D

Paul Speer, Ph.D

ACKNOWLEDGMENTS

Joining the PhD program at Vanderbilt turned out to be one of the most challenging, difficult and rewarding decisions of my life, and I am indebted to many people for supporting me throughout this journey.

To begin with, I would like to thank my advisor, Prof. Eugene Vorobeychik for accepting me as a graduate student in his lab, and encouraging (and demanding) me to pursue novel research. He gave me the freedom to learn, explore my own ideas and make mistakes, and always pushed me to aim for the best. It has been extremely fulfilling to have him as my doctoral advisor, and I hope to follow his ideas and principles if I create a lab of my own someday. I also learnt a lot while acting as a Teaching Assistant in his courses, and I am grateful to him for the opportunities. Eugene is surely one of the most influential people in my life, and I will always cherish the time that I spent under his guidance.

Next, I would like to thank the other members of my thesis committee. Prof. Abhishek Dubey, Prof. Maithilee Kunda, Prof. Paul Speer and Prof. Gautam Biswas have given me valuable insights and suggestions that have improved my thesis immensely. I have also collaborated, worked and published with Prof. Dubey, Prof. Speer and Prof. Biswas, and learnt a lot while working with them. A crucial part in my doctoral journey was played by all my collaborators; I would like to thank all of them, especially the Nashville Fire and Police Departments for working with me and guiding me throughout this journey.

One of the most enriching experiences at Vanderbilt has been my interaction with fellow students. I have collaborated with many graduate students, mentored under-graduate and high-school students and learnt a lot during such interactions. I am also extremely thankful to my fellow lab-mates at the CERL lab, who helped me settle down in an alien culture, helped me immensely in my research, and formed friendships that will hopefully stand the test of time.

The last year and a half of my graduate school was rather eventful. I lost my apartment and belongings due to a massive fire, and I couldn't be more thankful to an extremely close group of friends who took care of me during this troubled period. Moreover, my advisor moved to a different university during this time, making me unsure of the progress in my research. However, I am truly thankful to him for continuing to mentor, support and interact with me throughout this period, and for ensuring that I finished my dissertation in time. I also met my fiancée during this time, who loved and supported me through my highs and lows, and along with my close friends, helped me recover from a major knee surgery.

Finally, I am truly indebted to my family. I was attracted towards research largely due to the presence of successful scientists in my extended family, and they continue to inspire and motivate me. My parents made a lot of sacrifices to give me an incredibly happy childhood and excellent education. My mother gave up a career in law to raise two sons, and I could not be more thankful for everything that she has done. My father taught me the value of hard work, perseverance and principled thinking, and always encouraged me to achieve more. They instilled core principles and values in me, which have served me well throughout my life, and particularly during the difficult times of my doctoral work. My elder brother has been an immense influence in my life as well, and has always supported me and encouraged me to achieve my dreams, often through his own actions and choices. I could never have pursued a doctoral degree without the support of my family, and I am thankful to them for all the love and support.

Lastly, I would like to thank the Almighty for all the opportunities and gifts that have been bestowed upon me, and I am extremely grateful for them. I intend to use my skills and do my part, and hopefully make the world a slightly better place to live.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

Introduction

A constant threat that plagues humans across the globe today are incidents like fire, crime and traffic accidents. These concerns are further escalated by a rather alarming rise in population density across the globe (Nolan, 2004), making these incidents one of the major concerns for cities. Typically, these incidents are characterized by loss of life, damage to properties and injuries and are known as *emergencies*, which are defined as incidents that threaten public safety, health, and welfare. In 2016 alone, the United States of America witnessed over 18 million incidents of crime, over 6 million traffic accidents and over a million incidents concerning fire. The sheer number of such incidents dictates that significant efforts be made to deal with them effectively. Since these incidents affect the broader community to a great extent, the problem of addressing them engages governments at various levels. It is actually well-documented that one of the most important responsibilities of federal, state as well as local governments is mitigating and dealing with such events. As a result, it is incumbent upon governments to make plans, allocate resources and take preventive measures in order to alleviate threats that such incidents pose.

Before we look at how these plans are structured, it is helpful to get a high-level idea of the environment in which such incidents occur. In most urban areas, multiple types of incidents happen, and governments are equipped with different kinds of responders. Once such incidents happen, concerned organizations are notified and responders (possibly of different kinds, each with its specific responsibility) are dispatched to the scene of the incident. Responders must therefore be well prepared and strategically placed to attend to emergency calls. This entire system of planning, placing responders and dispatching them as and when incidents happen is what we refer to as the overall *emergency response system*. According to the Federal Emergency Management Agency, any well-designed approach to emergency response system entails the following four major stages - Mitigation, Preparedness, Response and Recovery. While all these stages are inter-linked, each presents its own implementational, technical as well as policy-level challenges. A further challenge is posed by the fact that these incidents are fundamentally different and hence, often times, it is difficult to design policies or algorithms that can cater to all incident types. For example, let us take the case of ambulances responding to traffic accidents and contrast that to police patrols responding to crimes - while ambulances must be located such that they reach accident sites as fast as possible, responding to a burglary that has already happened can be relatively slower. Another major difference is the effect of deterrence - a police patrol, by the sole virtue of its presence in an area is expected to dissipate or prevent crime; an ambulance however, cannot prevent the occurrence of accidents. This points out the need to be careful while dealing with these incident types, as neglecting these vital differences can cause models to fail to represent the real-world accurately. However, despite having differences, our research and experiments suggest that there exist technical commonalities among these problems. First, all such incidents happen in time and space and are caused by a set of environmental factors. While such factors differ for different types of incidents, the general structure of incident occurrence is the same. Secondly, all emergency responders work under the availability of limited resources, and thus a common problem is to allocate limited resources to respond to a set of spatial-temporal incidents. This existence of a *budget* and the spatial-temporal nature of incidents gives rise to similar problem structures for a variety of incidents. As an example, with an infinite number of police patrols, it should be possible in theory to protect all possible targets that can experience crimes. However, that is not the case, and this makes it necessary to understand how general patterns in incidents emerge, and deploy limited resources in accordance with such patterns, a problem that exists for all categories of incidents. This motivates us to create general-purpose algorithms and modeling approaches that can solve the core technical challenges involved, and then tailor these algorithms according to the needs of specific incident types.

These core technical challenges pose a variety of forecasting and decision-making problems, and our research has focused on understanding how incidents occur, how can models of occurrence be manipulated, how can resources be placed in anticipation of such incidents, and finally, how can algorithms be designed to respond to incidents. We provide a short sketch of the solution approach here; later, we introduce each of these problems separately over the course of this thesis, and explain how we aim to alleviate these problems.

In order to present a high-level overview of the solution approaches, we first define the problem at hand, albeit informally. The problem can be summarized as follows - given a variety of incidents in an urban area and a set of responders (possibly of different types), how do we ensure that responders *optimally* respond to incidents and if possible, deter or prevent them. Here, we take the liberty of not defining what *optimal* means formally, and we look at different definitions of effective response as we discuss specific problems later. Having described the broad question that we aim to ask, we now look at how we can break down the overall problem into atomic sub-problems. First, we point out that arguably, the most important cog in the wheel of any solution approach would be to understand how incidents occur - *what* causes these incidents, and *where* and *when* do they take place. Once spatial and temporal patterns emerge from historical data, we can then look at ways to allocate responders in anticipation of future incidents, and optimally dispatch responders as and when incidents happen. This segregation of the problem lets us look at the following sub-problems - 1) Prediction, 2) Placement, and 3) Dispatch. The structure of this thesis would revolve around this segregation. Throughout the rest of this text, we will aim to understand each of these problems, review existing literature around these problems from various domains, formulate the main problems formally, design scalable algorithms to solve these and compare and contrast our work with other state-of-the-art approaches.

We now shed some light on what approaches we take to tackle these problems, look at prior work in the area and point out how our work improves on the existing state-of-the-art algorithms. Incident Prediction, the most crucial part of this system, has received considerable attention in prior work (Murray et al., 2001; Levine et al., 2004; Short et al., 2008a; Mohler et al., 2011). Crime prediction, specifically, has seen significant efforts from various fields of research like statistics, operations research, computer science, mathematics and sociology. A common theme that binds these methods is the effort to capture spatial crime correlation, leading to the discovery of geographic areas known as "hotspots", and this continues to be one of the most used methods in practice. Two major shortcomings of this approach are its inability to capture arbitrary covariates that affect crime (like weather, proximity to pawn shops, liquor stores etc.) and the lack of a generative approach to modeling. In order to address these concerns, we propose using Survival Analysis to predict crimes, and later show that such models are flexible to work with other categories of incidents like accidents and fire. Originally developed in the area of medicine (Cox and Oakes, 1984), survival analysis methods essentially model risk or time to an event, which originally were deaths of patients. The models are extremely well-understood and popular in the medical as well as statistics community; a testament to this fact is that the seminal paper (Cox and Oakes, 1984) that introduced survival analysis is the most cited paper in the field of statistics. As far as its usage on predicting crimes is concerned, it is actually straight-forward to see why survival analysis methods are a natural fit to our problem. These methods can model time to incidents and capture arbitrary covariates naturally, alleviating both the problems mentioned above. We also show that a novel approach of combining hierarchical clustering (Johnson, 1967) with survival models results in prediction accuracy at least as good as existing state-of-the-art (often times better) approaches, and also causes a drastic reduction of run times. Another crucial issue with existing models is the inability to predict severity of incidents, which plays a crucial role in both placement and dispatch decisions. We approach this by modeling a joint density over incident arrival times and severity, and subsequently using conditional independence relationships to model time to arrival and incident severity by well-understood models (we discuss this in chapter III).

Forecasting incidents using historical data and a set of relevant features might not be enough, especially in the case of crimes. Unlike traffic accidents, crimes happen due to deliberate and planned actions, and criminals have direct incentives to deviate from prior patterns to evade detection. In order to capture this phenomenon, we explore the robustness of crime prediction models to account for adversarial manipulations. We frame this problem as a Stackelberg Game between a learner (police authorities) and an adversary (criminals) and formally capture how incident forecasting algorithms can be robust to such manipulations (we discuss this in chapter IV).

Armed with a generative prediction model that can capture arbitrary first-order effects on the occurrence of incidents, we next approach the problem of placing responders in anticipation of such events, which is generally a high-dimensional optimization problem under uncertainty. This problem must be dealt separately for different incident types since incidents being responded to have different needs (it is very important to send ambulances to the scene of an accident as fast as possible; a police patrol however, can balance secondary considerations while going to lodge a report for an incident that has already happened). Also, the arrival distributions of specific incident categories might actually be dependent on the distribution of responders (for example, the location of police patrols affects crimes). Hence, we deal with these problems separately. In order to optimally allocate police patrols, we propose a two-stage optimization problem formulation. In the first stage, police patrols are placed under uncertainty about crimes; and in the second stage, crimes occur and police responds to the crimes. We map the inner optimization problem to well-known linear optimization problems and use Bender's decomposition and sample-average approximation to solve the overall optimization problem. A key consideration, as pointed out, is the change in incident arrival distribution over arrival times with changes in police placements. To address this issue, we develop a novel iterative algorithm that repeatedly applied decomposition techniques to solve the optimization problem, while taking into account changes in the arrival distributions. Our experiments using real as well simulated data show that our decision-theoretic approach to dispatch police responders outperform existing strategies used in the field (we discuss this in chapter V).

Dispatching ambulances and fire trucks is a fundamentally different problem. Such responders usually work in extremely critical situations, and a matter of seconds could potentially be life-saving and dispatch decisions are usually taken in accordance with the severity or urgency of incidents. Also, ambulances and fire-trucks do not perform patrolling duties. Once response is finished, they usually return to concerned depots/stations (unless there are other pending calls). In order to place such responders optimally, we proposed a non-linear mathematical program to maximize the geographic coverage by responders, with constraints on the maximum waiting times according to different severities. The idea of maximizing coverage is self-explanatory - distribution of responders close to all areas that are likely to experience incidents is expected to decrease response times. Solving such a non-linear problem, with multiple severities is not trivial. In fact, approaches suggested in prior art, to the best of our knowledge, can only handle models in which all incidents are of equal severity, which severely limits the applicability of such approaches in practice. In order to address this, we design a novel algorithmic framework based on Greedy Random Adaptive Search (GRASP). Experiments using real-time traffic data from the Metropolitan area of Nashville showed significant reduction in wait-times for incidents (we discuss this in chapter VI).

Having created models to predict incidents and place responders, we now shift our focus to the problem that has the most applicability in practice - dispatching responders as and when incidents happen. While this problem is fairly well-studied in literature, most of the prior methods consider the environment in which such incidents happen to be *static*; the environment however, is *dynamic* and there are uncertainties associated with incident occurrence, travel times as well as service times. In order to bridge these gaps, we frame the dispatch problem as a Semi-Markov Decision Problem (SMDP) (Hu and Yue, 2007), that not only captures dynamics of a changing environment, but also removes strong (and impractical) distributional assumptions made on such models in prior art. This paradigm of modeling however, automatically results in a large and complicated state-space. We propose a Dynamic Bayes Network to leverage conditional independence relationships among state variables and provide a compact representation of state transition probabilities. This brings us to the most difficult technical problem associated with such a model - the state transition probabilities are unknown, difficult to model with well-known distributions. In order to alleviate this, we propose a novel approach - first, we transform the SMDP to a Discrete-Time Markov Decision Process (DTMDP). Then, we apply Policy Iteration with one crucial modification - we use simulations to estimate utilities of states and gradually learn statistically confident estimates of state-transition probabilities. It is crucial to point out that the learning happens while the decision-making problem is being solved, hence removing the need to explicitly learn or estimate the state transition probabilities separately. Once such estimates are available, the algorithm accesses the estimated probabilities and avoids simulation, thus saving time in terms of computation. We validate the algorithm using both real and simulated traffic accident data, and show significant reduction in response times over other state-of-the-art approaches (we discuss this in chapter VII).

The problems we have discussed thus far constitute the essential components of an emergency responder system. However, there are two crucial considerations that still need to be addressed to ensure that such approaches are deployed in practice - first, decisions considering response need to be taken very *quickly*, since time lost in calculating the optimal dispatch strategy could result in loss of life. Secondly, urban areas are dynamic environments, and the entire pipeline must be able to adapt to real-time changes in the environment. The SMDP model of dispatch, that we introduced before has an extremely large state space and involves learning statistically confident estimates of the state-transition probabilities. Once a policy is computed, finding the optimal action for any given state takes constant time. However, there can be changes in the environment that the original problem definition does not capture. For example, consider the case where a subset of the responders break down or the traffic conditions of the city changes suddenly. In such a scenario, the problem definition changes and even on a sufficiently *modern* computer, it would take days to re-train the model. In such scenarios, it might be worthwhile to create a system that can quickly find a good myopic action; such an action need not optimally minimize the expected response time but should perform well in the near future. We create a principled framework to tackle this problem. We use the same SMDP formulation as before, and draw ideas from Monte-Carlo Tree Search and Sparse Sampling of Markov Decision Processes to find an action, for a specific state of the system. The simulation provides an estimate of the *utility* associated with each action and lets us select one. We also discuss a methodology by which the survival model can be updated online with streaming data, which lets us update the incident forecasting ability as and when new incidents happen (we discuss this is chapter VIII).

While this area of research presents extremely interesting and thought-provoking technical challenges, it also presents the scope of real-world application. It is actually fair to claim that one of the most important yardsticks of judging such research is whether it can produce real-world impact or not. In order to facilitate this, we created a one-stop dashboard to allow end-users (firemen, fire-station managers, ambulance controllers and police personnel) to benefit from our research. Our research has extensively focused on collaborating with organizations that actually work in the field of emergency response. Over the past four years, we have collaborated with the Nashville Fire Department and the Metropolitan Nashville Police Department, and the invaluable domain expertise from these organizations has benefited our research immensely and provided us with means to validate our work. As a part of this thesis, we also describe in details the open-source dashboard that we are in the process of developing, and highlight how building such tools should be addressed (we discuss this in chapter IX).

We start by providing a detailed summary of prior work in the context of all the sub-problems we have identified in emergency response, and then look at each of the problems one by one.

CHAPTER II

Related Work

1 Incident Prediction

Incident prediction, resource allocation and dynamic dispatch of resources have attracted interest from various research communities over the years. Before we describe our algorithmic contributions in the following chapters, we first look at prior work in this domain, and identify gaps that need to be bridged.

There has been an extensive literature devoted to understanding and predicting incidents that can be collectively subsumed under the category of *emergencies*. Perhaps the maximum effort has been devoted to learn models of crime incidence, and this involves both qualitative and quantitative approaches. Crime models commonly fall into three categories: purely spatial models, which identify spatial features of previously observed crime, such as hot spots (or crime clusters), spatial-temporal models which attempt to capture dynamics of attractiveness of a discrete set of locations on a map, and risk-terrain models, which identify key environmental determinants (risk factors) of crime, and create an associated time-independent risk map.

Numerous efforts have been made to understand the occurrence of crimes in each of the above mentioned categories. A number of studies investigate the relationship between liquor outlets and crime (Speer et al., 1998; Toomey et al., 2012). Many of the earlier quantitative models of crime focus on capturing spatial crime correlation (hot spots), and make use of a number of statistical methods towards this end (Murray et al., 2001; Levine et al., 2004). Murray et al. (Murray et al., 2001) leveraged the use of geographic information systems and spatial analysis for examining crime occurrence, and such approaches are still the most commonly used methods in practice. An alternative approach, risk-terrain modeling, focuses on quantifiable environmental factors as determinants of spatial crime incidence, rather than looking at crime correlation (Kennedy et al., 2011). Risk Terrain Modeling emerged as a tool to identify behavioral settings for a specific jurisdiction that resulted in crimes, and has been adopted by many law enforcement agencies to combat crimes. However, both these classes of models have a key limitation: they ignore the temporal dynamics of crime. Moreover, environmental risk factors and spatial crime analysis are likely complementary. Our approach aims to merge these ideas in a principled way.

Recently, a number of sophisticated modeling approaches emerged aiming to tackle the full spatio-temporal complexity of crime dynamics. One of these is based on a spatio-temporal differential equation model that captures both spatial and temporal crime correlation (Short et al., 2008a; Mohler et al., 2011). These models have two disadvantages compared to ours: first, they do not naturally capture crime co-variates, and second, they are non-trivial to learn from data (Mohler et al., 2011), as well as to use in making predictions. Another model in this general paradigm is Dynamic Spatial Disaggregation Approach (DSDA) (Ivaha et al., 2007), which combines an autoregressive model to capture temporal crime patterns with spatial clustering techniques to model spatial correlations. The model we propose is significantly more flexible, and combines spatial and temporal predictions in a principled way by using well-understood survival analysis methods. Recently, an approach has been proposed for modeling spatial and temporal crime dynamics using Dynamic Bayes Networks (Zhang et al., 2015, 2016). This approach necessitates discretization of time, as well as space. Moreover, despite significant recent advances, scalability of this framework remains a challenge.

A somewhat orthogonal approach to crime prediction seeks to model the interaction between law enforcement agencies and criminals as *games*. Specifically, the paradigm of Stackelberg Games has been extensively used in the context of security domains. The particular notion that makes the paradigm relevant is its incorporation of a leader-follower model (Von Stengel and Zamir, 2004). In Stackelberg Security Games, a defender is given the responsibility to defend a set of targets using a limited number of resources; the attacker, on the other hand, is given the chance to observe the strategy undertaken by the defender and plan accordingly. The game is typically modeled for a single round. Such a model has been used to deploy air marshals to flights (Tsai et al., 2009), protect biodiversity in conservation areas (Fang et al., 2016), and screen passengers in airports in USA (Brown et al., 2016). An extension to this paradigm was made by the introduction of Green Security Games, that model the repeated interaction between criminals and law enforcement agencies by extending the leader-follower paradigm to multiple rounds (Haskell et al., 2014; Johnson et al., 2012). In such games, attacker behavior in previous rounds can be used to better policing decisions in subsequent rounds.

Other categories of incidents have also been looked at. There has been significant effort devoted to predicting freeway accidents (Songchitruksa and Balke, 2006; Ackaah and Salifu, 2011), traffic congestion prediction (Balke et al., 2005), fire accident predictions (Zhanli et al., 2010), and many more. Predicting accidents is a harder problem however, and it has been noted in the literature that freeway accidents are generally difficult to predict, due to an inherent random nature of these accidents and spatially varying factors (Qi et al., 2007). Recently, freeway accidents have been predicted using panel data analysis approach that predicts incidents based on both time-varying and site-specific factors (Qi et al., 2007). An extensive survey of the literature on crash prediction models is presented by Kiattikomol (Kiattikomol, 2005), which highlights the prevalence of Poisson distribution based models (Bonneson and McCoy, 1993; Maher and Summersgill, 1996; Sayed and Rodriguez, 1999), and multiple linear regression approaches (Frantzeskakis et al., 1994; Resende and Benekohal, 1997). These approaches treat accidents as homogeneous, meaning that the severity of accidents is not taken into account, which is a crucial factor in practice. Another crucial limitation in prior art is the absence of a principled approach to learn the spatial granularity at which incident prediction models should be learned.

In chapter III, we bridge these shortcomings in prior art by considering the problem of incident prediction by looking at both spatial and temporal dynamics of incident occurrence, as well as its severity. Also, our model is flexible enough to accommodate arbitrary features naturally, allowing us to capture covariates that affect incidents. In order to learn the spatial granularity at which these models can be learned, we predict a novel approach that combines survival analysis with hierarchical clustering (Johnson, 1967) to balance spatial heterogeneity and model variance.

While incident forecasting is crucial to any pipeline for emergency response, there are finer considerations that must be taken into account. For example, crime prediction solely based on historical data and environmental factors ignores a crucial factor - criminals can gain knowledge of policing strategies and adapt themselves, thus rendering predictive models stale and inaccurate (Reppetto, 1976). A major shortcoming of many prior predictive approaches is that they do not account for such adversarial adaptation. We tackle this problem in chapter IV, by modeling the interaction between law enforcement authorities (defender) and criminals (attackers)attacker as a game, and accounting for attacker manipulation while learning a model for the defender. Such a problem falls under the broader paradigm of adversarial learning, which looks to study the effect of adversarial influence on machine learning models (Barreno et al., 2006; Huang et al., 2011; Vorobeychik, 2016). Adversarial Learning has been successfully used in many domains of machine learning, and recently, it has been used to combat some implicit biases and imperfections in crime prediction methodologies as well (Gholami et al., 2018). However, to the best our knowledge, models specifically looking to create robust crime prediction approaches have not been explored. We specifically focus on the problem of criminals manipulating their behavior in response to a specific learned model for crime prediction, and create a framework for predicting crimes that is robust to such manipulations.

2 Responder Allocation

The problem of optimally placing responders in space to respond to incidents has been well explored in literature. As an important first step, we recognize the fact that there are multiple measures of optimality in this context. The most natural measure of the quality of response is response time, and considerable research has been devoted to this (Zografos et al., 1993, 2002). Another common criterion is to maximize the coverage area of response vehicles (Majzoubi, 2014). It is also natural to combine these goals and create allocation algorithms that try to achieve both (Silva and Serra, 2008; Eaton et al., 1985). An obvious way to formulate such a problem is to include one goal as the objective in the resulting optimization problems and specifying the others as constraints to achieve a desired level of service.

We focus specifically on coverage models, that aim to maximize the reach of service providers to potential demand nodes. In particular, the Maximum Coverage Problem (Church and Velle, 1974) looks at optimally locating p facilities such that the maximum number of demand nodes can be served. The problem of locating facilities on a network to minimize travel distance to potential nodes was actually formulated before this (Hakimi, 1964). The problem of minimizing the average service time from a single server location where requests are queued in the absence of a server has also been explored before (Berman et al., 1985). The same problem, with service time distributed negative exponential, has also been explored and it was proved that optimal location of server corresponds to the corresponding transportation network for the problem (Berman and Larson, 1982). Another approach is to look at minimizing the expected response time to the furthest point in the network from a single server location (Brandeau and Chiu, 1992). A common issue with these approaches is that "responder *depots*" and "responders" are used synonymously, meaning that only one responder is placed at a location. In practice, this is typically not the case, severely limiting the ability to apply such methods in the field. Responder services typically rent or own spaces that can house multiple responders, which calls for the dual optimization over depot locations as well as the number of responders per depot. We extend prior work concerning maximizing coverage with restrictions on waiting times for incidents (Silva and Serra, 2008) to tackle this dual optimization scenario, thereby bridging a major gap in prior literature.

3 Responder Dispatch

The problem of optimally dispatching responders as incidents happen has intrigued researchers for long. The simplest case of the problem is the well-known transportation problem (Munkres, 1957), which given a fixed set of incidents and responders, can find the path that should be travelled by each responder such that the overall distance traveled by all the responders is minimized. The problem of responding to incidents that constitute as emergencies is different. The set of incidents that must be responded to are not known beforehand. Incident calls come one at a time, and decisions regarding dispatch must be taken in real-time to minimize the response times to all incidents. This problem has also been looked at in literature, and it has been noted as a non-trivial problem particularly when multiple responder types are involved (Li et al., 2011). Also, emergency response has been noted to face inherent uncertainty, as although consumers do not know beforehand when the service is needed, there is an obvious expectation of quick response once a call is made (Felder and Brinkmann, 2002). This problem has typically been studied as part of the responder location problem (Li et al., 2011) as well as a joint optimization problem of fairly distributing resources and optimizing response times (Toro-DíAz et al., 2013). Most of the approaches mentioned above, to either the responder location, or dispatch, are *static*, given a particular distribution over incidents in space and time. The environment, however, is dynamic, and allowing dispatch to adjust to current information and environment is a crucial consideration in practice which is generally ignored in prior art. A recent approach has taken a decision-theoretic approach to dispatching responders and is quite principled and well-structured. It however, suffers from a major flaw - it assumes that the total response times for emergency responders (sum of travel time and service time) are exponentially distributed. This strong distributional assumption results in two simplifications - first, it lets the usage of well-known distributions to model and estimate the state transition probabilities in closed-form and secondly, enables the usage of Continuous-Time Markov Decision Processes. However, travel times are not exponentially distributed in practice, and this limits the practical applicability of the work. We try to address these problems systematically in chapter VII. Our approach, without such assumptions creates a decision-theoretic framework that is practically applicable (we only assume Poisson distributed incident arrivals and exponentially distributed response times, both of which are actually well-established models that are used extensively in prior work). We achieve this by using the more general class of Semi-Markov Decision Processes. However, this modeling paradigm introduces additional technical challenges that are non-trivial; we highlight these challenges and ways to address them.

CHAPTER III

Incident Prediction

1 Introduction

The most fundamental problem in designing approaches to manage emergency response is to understand how such incidents occur in time and space. This motivates us to design ways to learn spatial temporal models of incident occurrence. At first glance, the title of this chapter seems to be a more difficult problem than the problem description we just stated - indeed, an understanding of how incidents happen does not necessarily provide an instrument to predict such incidents. However, it is essential to remember that the broader goal of developing an effective emergency response system is to solve the problem of efficiently placing responders and dispatching them as incidents happen. One way to design such a system is to create capabilities of simulating the *world* under consideration, and then learn from such simulations. This in turn, makes it imperative that we design models that let us forecast incidents. The infrastructure to simulate the world lets algorithms internally learn and understand which decisions are better, and helps us validate the choices made by the algorithms we design. Another important consideration is that often times, understanding when and where incidents happen does not suffice; and one might want to learn about what covariates affect and cause such incidents (we study only correlation and do claim any results about causality; causality is used here only to motivate the approach). This helps us in two ways - first, it lets us create more accurate models by capturing first-order effects that influence incidents and secondly, it paves the way for making better policy decisions. As an example, understanding how speed limits affect traffic accidents or how liquor stores affect crimes in a particular city can lead to better policy decisions about zoning and hours of operation for liquor stores and speed limits, and will make our approaches more robust to environmental changes that affect such variables (such as changes in the weather or the opening of new pawn shops in an area). Hence, we focus on generative models of incident prediction. Through the rest of the thesis, models of incident arrival would implicitly refer to generative models which, given a set of arbitrary features, provides us the means to sample incidents and therefore act as a simulator.

2 Continuous-Time Crime Forecasting

We start by looking at a formal model that can forecast incidents in time and space, and then look at how severity of incidents can be modeled.

2.1 Predicting incident arrival

2.1.1 Temporal Dynamics

We consider a dataset D of crime incidents, in which incident $d_i \in D$ denotes the i^{th} incident. We assume that the spatial granularity for any learning model dealing with this area is captured by the grids. Consequently, it suffices to denote the spatial location of each incident by one of the grids. Further, each incident $d_i \in D$ is also characterized by a time of occurrence x_i and a set of spatial-temporal features $w_i \in \mathbb{R}^m$. Thus, our dataset is a collection of time-stamped feature vectors corresponding to crime incidents, $\{(x_1, w_1), (x_2, w_2), \ldots, (x_n, w_n)\}$. For each grid $g_i \in G$, we denote the time between successive incidents by the random variable τ, and define $\tau_i = x_i - t_{x-1}$ as the time to arrival of the ith incident in the dataset. Let the time between a specific incident type (crime, fire, accidents and so on) be represented by the random variable t. Also, let us assume that the entire spatial area in consideration is divided into a set of discrete set of equally sized grids G. We propose to learn a density $f(t|w)$ over time to arrival for this set of discrete spatial locations, where w is a set of arbitrary covariates.

A natural choice for this problem is survival analysis (Cox and Oakes, 1984) which allows us to represent distribution of time to events as a function of arbitrary features. Formally, the survival model is $f_t(t|\gamma(w))$, where f_t is a probability distribution for a continuous random variable T representing the inter-arrival time, which typically depends on covariates w as $\log(\gamma(w)) = \rho_0 + \sum_i \rho_i w_i$. A key component in a survival model is the survival function, which is defined as $S(t) = 1 - F_t(t)$, where $F_t(t)$ is the cumulative distribution function of T. Survival models can be parametric or non-parametric in nature, with parametric models assuming that *survival time* follows a known distribution. In order to model and learn $f(t)$ and consequently $S(t)$, we chose the exponential distribution, which has been widely used to model inter-arrival time to events and has the important property of being memory-less, the use of which we explain shortly (it is also particularly useful when creating the responder dispatch model; we highlight this in Chapter 5). We use Accelerated Failure Model (AFT) for the survival function over the semi-parametric Cox's proportional hazard model (PHM) and estimate the model coefficients using maximum likelihood estimation (MLE), such that in our setting, $S(t|\gamma(w)) = S(\gamma(w)\, t)$. While both the AFT and PHM models measure the effects of the given covariates, the former measures it with respect to survival time and the latter does so with respect to the hazard. The AFT model thus allows us to offer natural interpretations regarding how covariates affect crime rate.

A potential concern in using survival analysis in this setting is that grids can experience multiple events. We deal with this by learning and interpreting the model in a way that the multiple events in a particular grid are treated as single events from multiple grids and prior events are taken into consideration by updating the temporal and spatial covariates.

We come back to the choice of an exponential distribution to model and learn $f(t)$. Widely used to model arrival times, the exponential distribution is the most natural choice for our model. Events, conditional on a set of features, are sampled i.i.d from this distribution. While we describe the covariates that affect incident occurrence later, we highlight here that one of the most important features are spatial correlation with historical incidents. Therefore, every incident occurrence affects future events. Thus, for a discrete location, as we wait for a crime to happen, the waiting time gets affected by events happening in other locations. The exponential distribution is particularly useful here since it is memory-less. When an event happens, the waiting times for all other discrete locations are reset, and their respective feature vectors are potentially updated due to the said event. Then, the next event is sampled for all locations.

Originally in the field of medicine, each point of data used in survival model came from different specific patient or subject. Our model is different. Grids can experience multiple events, and therefore we consider that each spatial grid introduces a new "subject" every time an event happens. This can be detrimental as the effect of earlier events on future events can easily be ignored under such a setting (e.g. a grid that has experienced events recently is more likely to experience another event soon, and is thus not a "new subject" per se). Our model, however, deals with this by accommodating such information in the feature space - with every event, grids have an updated set of temporal and spatial covariates that account for this correlation.

2.1.2 Spatial Dynamics

In learning the survival model above, there is a range of choices about its spatial granularity, from a single homogeneous model which captures spatial heterogeneity entirely through the model parameters w, to a collection of distinct models f_i for each spatial grid $i \in G$.

A natural way to capture the incidents in space is to first discretize space into areas as described before, and then learn survival models independently for each grid. The main concern with this approach is overfitting: each grid induces relatively little data, and there are surely considerable structural similarities of the incident process across multiple grids that we can leverage. On the other hand, learning a single "universal" model for all grids may fail to capture all of the existing heterogeneity not explicitly modeled in the feature space w.

We propose two methods to solve this problem. In the first (Mukhopadhyay et al., 2016a), we split the discrete spatial areas into two coarse categories: grids that experience a high number of incidents and grids that experience low number of incidents, and learned two distinct homogeneous models for these. We do this by treating the count of crimes for each grid as a data point and then splitting the data into two clusters using *k-means* clustering. We refer to this process as the basic Probabilistic Survival Model (PSM).

However, the number of clusters in this approach is not learned from data and the coarse categorization is fixed. In order to address this, we propose a combination of Hierarchical Clustering approach (Johnson, 1967) with survival analysis. In order to explain how this works, we first introduce some notation. For an incident, let the feature set w be divided into two parts w_s and w_d, where w_s represents a set of *static* features, such as population density in a grid, which remain relatively stable, while w_d will denote *dynamic* features, such as the amount of rainfall in a day or day of the week. We hypothesize that the set w_s can be used to identify similarity between distinct spatial grids. To operationalize this hypothesis, we propose a hierarchical clustering algorithm, shown in Algorithm 1 which we now describe at a high level. We start by treating each grid as a distinct cluster. Iteratively, we merge two grids that are most similar, with similarity between grid i and grid j measured as the distance between associated w_s^i and w_s^j. At each step, we check whether the updated set of clusters decreases the predicted likelihood computed on the training data set compared with the previous iteration by more than a pre-defined limit, and stop as soon as marginal improvement in likelihood is below this limit. We maintain a high likelihood difference tolerance level initially to promote exploration of the solution space, and lower it as the algorithm progresses.

Algorithm 1 Hierarchical Clustering

```
INPUT: Grids G, Static Features W_s, Likelihood Model L
OUTPUT: Clusters C, with optimal likelihood
Initialize C^0 ← each grid g_i as a cluster.
for iteration m in max_iter do
    Calculate Similarity Matrix S, where S_{i,j} = ||w_s^i − w_s^j||
    i, j = argmin_{ij} S
    Merge c^i, c^j into c^i
    Update w_s^i = (w_s^i + w_s^i) / 2
    Calculate Likelihood L^m
    if L^m − L^{m−1} > σ/m then
        Return C^m
    end if
    Return C^m
end for
```

2.2 Predicting Incident Severity

Predicting time to arrival for incidents is crucial, but treating all incidents as identical, as is commonly done, is problematic in practice. As an example, consider two grids g_1 and g_2 with similar rates of predicted traffic incidents with one major difference: most incidents that happen in g_1 require immediate medical attention while most incidents in g_2 are minor accidents. Focusing solely on incident rates to allocate medical response vehicles would clearly be unwise from the perspective of saving lives. Consequently, it is imperative that we also predict the severity of an incident or the urgency with which it needs a response. Here, we point out that dispatching emergency response based on predictions is undesirable, as real-time information must be taken into account for accurate severity assessment. However, planning aggregate depot and responder locations necessitates predicting severities.

One way to capture incident severity is to use a distinct model for each incident type. However, past incident prediction models (Gorr et al., 2003) have found that sacrificing the scale of data for achieving heterogeneity can produce noisy estimates, as it limits the data available for learning each distribution. We address this issue by learning a joint distribution over arrival time t and incident severity k, $f(t,k|w)$, where k is a discrete ordinal random variable representing the severity class of the incident from K possibilities. As a first, step, we represent $f(t,k|w) = f(t|w)f(k|t,w)$. This decomposition helps us in two ways: first, our model for predicting arrival times described in Section 2.1 can now be used as is to learn the density over arrival times, and second, we can now use the entire dataset to learn distribution over arrival times and severities, rather than fracturing it by severity category. To learn the severity distribution (conditional on incident time and the feature vector w) $f(k|t,w)$, we use the multinomial logistic regression model (Böhning, 1992).

$$\begin{bmatrix} P(y=1|x,\theta) \\ P(y=2|x,\theta) \\ . \\ . \\ P(y=k|x,\theta) \end{bmatrix} = \begin{bmatrix} \frac{e^{\theta_1^T x}}{\sum_{j=1}^{K} e^{\theta_j^T} x} \\ \frac{e^{\theta_2^T x}}{\sum_{j=1}^{K} e^{\theta_j^T} x}) \\ . \\ . \\ \frac{e^{\theta_3^T x}}{\sum_{j=1}^{K} e^{\theta_j^T} x} \end{bmatrix} \tag{III.1}$$

Multinomial logistic regression (MLR) generalizes the standard logistic regression by extending the output variable to a general categorical variable.

Formally, given a training set $\{(w^1,t^1,y^1),(w^2,t^2,y^2),...,(w^n,t^n,y^n),\}$, where $w^i \in \mathbb{R}^m$, $t \in \mathbb{R}$ and

$y^i \in 1,2...,K$, MLR models $P(y=k|w,t)\,\forall k \in 1,2,...,K$. The hypothesis function in this case can be represented as shown in Eq. (III.1) where we represent the set $\{w,t\}$ by a generic feature vector x. The cost function that we try to minimize is:

$$J(\theta) = \sum_{i=1}^{m}\sum_{k=1}^{K} \mathbb{1}\{y^i = k\} \log \frac{e^{\theta_k^T x_i}}{\sum_{j=1}^{K} e^{\theta_j^T x_i}}$$

The cost function is typically minimized by an iterative optimization algorithm such as gradient descent.

The next step in the modeling process is to identify a collection of features that impact incidents, which will comprise the co-variate vector w. For obvious reasons, we define the feature sets differently for crimes and accidents.

3 Features

3.1 Features affecting crime

Our experiments suggest that crimes are affected by a variety of covariates. To present the features in a meaningful and intuitive manner, we divide them into temporal (those that only change with time), spatial (those capturing spatial heterogeneity), and spatio-temporal (features changing with both time and space).

3.1.1 Temporal Features

Temporal Crime Cycles: Preliminary analysis and prior work (Felson and Poulsen, 2003; Landau and Fridman, 1993) were used to identify the set of covariates, such as daily, weekly and seasonal cycles, that affect crime rate. Crime rates have also been shown to depend on seasons (with more crime generally occurring in the summer) (Lauritsen and White, 2014). Thus, we consider seasons as binary features. In order to incorporate crime variation throughout the day, each day was divided into six zones of four hours each, captured as binary features. Similarly, another binary feature was used to encode weekdays and weekends.

Temporal Crime Correlation: It has previously been observed that crime exhibits inter-temporal correlation (that is, more recent crime incidents increase the likelihood of subsequent crime). To capture this aspect, we used recent crime counts in the week and month preceding time under consideration.

Weather: It is known that weather patterns can have a significant effect on crime incidence (Cohn, 1990). Consequently, we included a collection of weather-related features, such as rainfall, snowfall, and mean temperature.

Police Presence: The final class of features that are particularly pertinent to our optimization problem involves the effect of police presence on crimes. Specifically, it is often hypothesized that police presence at or near a location will affect future crime at that location Koper (1995). We try to capture this relationship, by including a feature in the model corresponding to the number of police vehicles passing within the grid, as well as its immediate neighboring grid cells, over the previous two hours.

3.1.2 Spatial and Spatio-Temporal Features

Risk-Terrain Features: We leveraged the risk-terrain modeling framework (Kennedy et al., 2011), as well as domain experts, to develop a collection of spatial features such as population density, mean household income, and housing density at the census tract level. We used the location of pawn shops, homeless shelters, liquor stores, and retail outlets that sell liquor as the observed spatial-temporal variables (note that temporal variation is introduced, for example, as new shops open or close down).

Spatial Crime Correlation: One of the most widely cited features of crime is its spatial correlation (also referred to as *repeat victimization* (Kleemans, 2001)), a phenomenon commonly captured in hot-spotting or spatial crime clustering techniques. We capture spatial correlation as follows. For each discrete grid cell in the space we first consider the number of crime incidents over the past two days, past week, and past month, as model features, capturing repeat victimization within the same area. In addition, we capture the same features of past crime incidents for neighboring grid cells, capturing spatial correlation.

Spatial Effects of Police Presence: Aside from the temporal effect of police on crime (reducing its frequency at a particular grid cell), there is also a spatial effect. Specifically, in many cases criminals may simply commit crime elsewhere (Hope, 1994). To capture this effect, we assume that the spillover of crime will occur between relatively nearby grid cells. Consequently, we add features which measure the number of police patrol units over the previous two hours in grid cells that are not immediately adjacent, but are several grid cells apart. In effect, for a grid cell, we hypothesize that cells that are very close push crime away or reduce it, whereas farther away grids spatially shift crime to the concerned grid, causing spillover effects.

3.2 Features affecting accidents

The next step in this model of incident prediction is to select a set of features w that can be used to learn the predictive model for traffic accidents. We describe here the features that constitute the set w. **Temporal Cycles** We used preliminary analysis and prior work in incident prediction to identify the different types of seasonality that affect incidents in our dataset. We use a binary feature for each season and another one for encoding weekdays and weekends. In order to look at the effect of time of day on incidents, we split each day into six zones of four hours each, and captured these by binary features.

Temporal and Spatial Incident Correlation For each grid, we looked at the past incident counts in the last week and month in it as well as neighboring grids as features to capture the effect of temporal and spatial correlation among incidents. We also treated the number of past incidents in each severity category as a feature while predicting incident severity, and considered the long-term effect of temporal correlation by looking at the average number of incidents in the past year.

Weather It is known that weather affects traffic incidents (Songchitruksa and Balke, 2006). We included a collection of features, such as rainfall, snowfall, and mean temperature to capture this effect.

Transportation Features The effect of roadway geometry on accidents has been extensively studied (Poch and Mannering, 1996), (Shankar et al., 1995). For each grid, we used the total number of roadway and highway intersections as features.

We use average incident counts in a grid and the set of transportation features as the set of static features w_s while the other features form the set of dynamic features w_d.

Having described the model and the features, we now describe the data that we use for validating our algorithms and hypotheses.

4 Data

Our evaluation uses crime as well as traffic accident data obtained from the *police and fire department* in Nashville, USA, with a population of approximately 700,000. The crime data we obtained is composed of burglary data from 2009 for Davidson County, TN, a total of 4,627 incidents, which includes coordinates and reported occurrence times. Observations that lacked coordinates were geo-coded from their addresses. In addition, we used police vehicle patrol data for the same county, consisting of GPS dispatches sent by county police vehicles, for a total of 31,481,268 data points, where each point consists of a unique vehicle ID, time, and spatial coordinates. A total of 624 retail shops that sell liquor, 2494 liquor outlets, 41 homeless shelters, and 52 pawn shops were taken into account. We considered weather data collected at the county level. Additional risk-terrain features, included population density, housing density, and mean household income at a census tracts level.

For the fire department of Nashville, traffic accidents comprise a large majority of incidents it responds to (fires, in contrast, are relatively rare). We looked at data for 26 months, from 2014 - 2016, comprising of a total of 20148 traffic accidents. Each accident is accompanied by its time of occurrence, the time at which the first responding vehicle reached the scene and the time at which the last responding vehicle was back at service, which refers to completion of servicing an incident. To predict incidents, we extracted highway and street intersections from Open Street Maps (Haklay and Weber, 2008) and weather data was collected at the county level.

5 Results

5.1 Predicting Crimes

The first result we present is the evaluation of our proposed continuous-time model based on survival analysis to forecast crime. Our parametric model is simpler (in most cases, significantly) than state-of-the-art alternatives, and can be learned using standard maximum likelihood methods for learning survival models. Moreover, it is nearly homogeneous: only two distinct such models are learned, one for low-crime regions, and another for high-crime regions. This offers a significant advantage both in interpretability of the model itself, as well as ease of use. Moreover, because our model incorporates environmental factors, such as locations of pawn shops and liquor stores, it can be naturally adapted to situations in which these change (for example, pawn shops closing down), enabling use in policy decisions besides police patrolling. On the other hand, one may expect that such a model would result in significant degradation in prediction efficacy compared to models which allow low-resolution spatial heterogeneity. As we show below, remarkably, our model actually outperforms alternatives both in terms of prediction efficacy, and, rather dramatically, in terms of running time.

For this evaluation, we divided our data into 3 overlapping datasets, each of 7 months. For each dataset, we used 6 months of data as our training set and 1 month's data as the test set. For spatial discretization, we use square grids of sides 1 mile throughout, creating a total of 900 grids for the entire area under consideration. While our model is continuous-time, we draw a comparison to both a continuous-time and a discrete-time models in prior art. However, since these are not directly comparable, we deal with each separately, starting with the continuous-time DSDA model. We refer to the DSDA model simply as *DSDA*, the model based on a Dynamic Bayes Network is termed *DBN*, and our model is referred to as *PSM* (parametric survival model).

5.1.1 Prediction Effectiveness Comparison with DSDA

Our first experiments involve a direct performance comparison to a state-of-the-art DSDA model due to Ihava et al. (Ivaha et al., 2007). We chose this model for two reasons. First, DSDA provides a platform to make a direct comparison to a continuous time model. Second, it uses time series modeling and CrimeStat, both widely used tools in temporal and spatial crime analysis.

We introduce the underlying concept of the model before comparing our results. DSDA segregates temporal and spatial aspects of crime prediction and learns them separately. In the temporal model, days like Christmas, Halloween, and football match days that are expected to show deviation from the usual crime trend are modeled using hierarchical profiling (HPA) by using the complement of the gamma function:

$$y = a_p - b_p t^{c_p - 1} e^{-d_p t}$$

where y is observed count, t is time and a_p, b_p, c_p and d_p are the parameters to be estimated using ordinary least squares (OLS).

All other days are initially assumed to be part of a usual average weekly crime rate, which is modeled using the following harmonic function

$$y = a_a - b_a t + c_a t^2 + \sum_{i=1}^{26} \left[d_a \cos\left(\frac{i\pi t}{26}\right) + e_a \sin\left(\frac{i\pi t}{26}\right) \right]$$

where y_a is the weekly crime average, t is time and a_a, b_a, c_a, d_a and e_a are the parameters that are estimated using OLS. Then, the deviations are calculated from the observed data and these are again modeled using the harmonic function. This forms the deterministic part of the model $f(t)$. The error Z from the observed data is modeled using seasonal ARIMA, and the final model is $y = f(t) + Z$. The spatial component of DSDA was evaluated using STAC (Ivaha et al., 2007), which is now a part of CrimeStat (Levine et al., 2004).

In order to make a comparative analysis, we considered a natural adaptation of the HPA-STAC model, which enables us to compare likelihoods. We use the outputs (counts of crime) from the HPA model as a mean of a Poisson random variable, and sample the number of crimes from this distribution for each day. For the spatial model, HPA-STAC outputs weighted clusters in the form of standard deviation ellipses, a technique used commonly in crime prediction. Here, we consider that:

$$P(x_i) = P(c(x_i)) P(x_i^{c(x_i)})$$

where $P(x_i)$ is the likelihood of a crime happening at a spatial point x_i which belongs to cluster c_i, $P(c(x_i))$ is the probability of choosing the cluster to which point x_i belongs from the set of all clusters and $P(x_i^{c(x_i)})$ is the probability of choosing point x_i from its cluster c_i. We assume that $P(x_i^{c_i}) \propto \frac{1}{Area_{c(x_i)}}$. Finally, we assume that the total likelihood is proportional to the product of the spatial and temporal likelihoods.

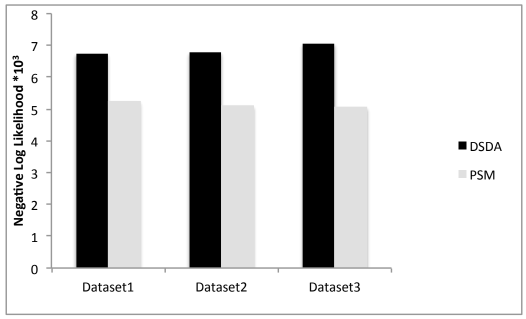


Figure III.1: Likelihood comparison of PSM vs DSDA.

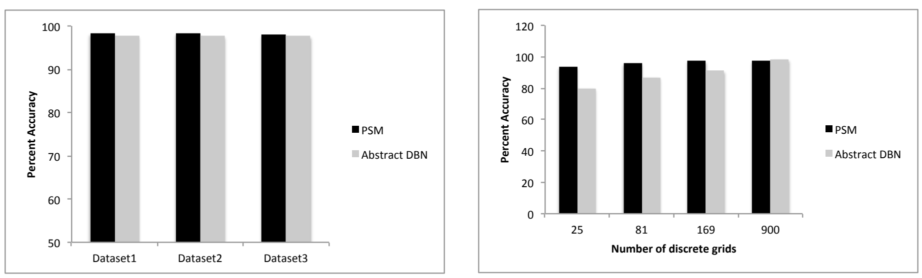
Figure III.1 shows the comparison of DSDA log-likelihood (on test data) for the three datasets described above. Indeed, our model outperforms DSDA in both the temporal and the spatial predictions by a large margin (overall, the improvement in log-likelihood is 25-30%).

5.1.2 Prediction Effectiveness Comparison with the Dynamic Bayes-Network Model

Next, we compare our model to the framework proposed by Zhang et al. (Zhang et al., 2015), which looks at crime prediction by learning a non-parametric Dynamic Bayes Network (DBN) representation, and applying abstraction techniques to improve scalability (Zhang et al., 2016). The DBN includes three sets of state variables: numbers of police vehicles in each grid i at time t, denoted by D_{it}, the number of criminals in grid i at time t, X_{it}, and the number of crimes Y_{it} in each grid i at time t. The main assumptions of this DBN are that a) police vehicle dynamics are known (so they are not random variables), b) locations of criminals at time $t+1$ only depends on patrol and criminal (but not crime) locations at time t, and c) crime incidents at time t only depend on locations of criminals and police at time t. Consequently, the problem involves learning two sets of transition models: $P(X_{i,t+1}|D_{1,t},...,D_{N,t},X_{1,t},...,X_{N,t})$ and $P(Y_{i,t}|D_{1,t},...,D_{N,t},X_{1,t},...,X_{N,t})$ for all grid cells i, which are assumed to be independent of time t. Since the model involves hidden variables X, Zhang et al. learn it using the Expectation-Maximization framework. While the model is quite general, Zhang et al. treat X, Y, and D as binary.

Since our proposed model is continuous-time, whereas Zhang et al. model is in discrete-time, we transform our model forecasts into a single probability of at least one crime event occurring in the corresponding interval. Specifically, we break time into 8-hour intervals (same temporal discretization as used by Zhang et al.), and derive the conditional likelihood of observed crime as follows. Given our distribution $f(t|w)$ over inter-arrival times of crimes, and a given time interval $[t_1, t_2]$, we calculate the probability of observing a crime in the interval as $F(t \leq t_2|w) - F(t \leq t_1|w)$, where F represents the corresponding cumulative distribution function (cdf).

To draw the most fair comparison to DBN, we use an evaluation metric proposed by Zhang et al. (Zhang et al., 2016) which is referred to as *accuracy*. Accuracy is calculated as a measure of correct predictions made for each grid and each time-step. For example, if the model predicts a probability of crime as 60% for a target, and the target experiences a crime, then the accuracy is incremented by 0.6. Formally, let p_i be the predicted likelihood of observing a crime count for data point i. Then accuracy is defined as $\frac{1}{m}\sum_i p_i$, where i ranges over the discrete-time sequence of crime counts across time and grids and m the total number of such time-grid items.



(a) (b)

Figure III.2: Accuracy comparison between PSM and Abstract DBN. (a) Varying data subsets. (b) Varying the number of grids.

Figure III.2(a) shows the results of accuracy comparison (with the accuracy measure defined above) between the DBN model and our model (PSM). We can observe that both models perform extremely well on the accuracy measure, with our model very slightly outperforming DBN. We also make comparisons by varying the number of grids, shown in Figure III.2 (b), starting around downtown Nashville and gradually moving outwards. Our model outperforms DBN in all but one case, in which the accuracies are almost identical.

5.1.3 Runtime Comparison with DSDA and DBN

We already saw that our PSM model, despite its marked simplicity, outperforms two state-of-the-art forecasting models, representing continuous-time and discrete-time prediction methods, in terms of prediction efficacy. An arguably more important technical advantage of PSM over these is running time. Figure III.3 shows running times (of training) for PSM, DSDA, and DBN (using the

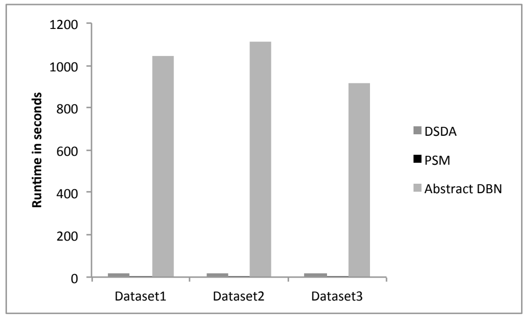


Figure III.3: Runtime comparison (seconds) between DSDA, Abstract DBN, and PSM.

abstraction scheme proposed by Zhang et al. Zhang et al. (2016)). The DBN framework is significantly slower than both DSDA and PSM. Indeed, PSM running time is so small by comparison to both DSDA and DBN that it is nearly invisible on this plot.

5.2 Predicting Accidents

Having evaluated the performance of the proposed models on crime data, we now evaluate the performance of our incident prediction model on accident data. Here, we compare the combination of hierarchical survival analysis with CCSA, and show how learning the degree of spatial clustering from data helps us achieve higher likelihood in test data. To evaluate incident prediction performance, we split our data into three overlapping data sets of 22 months each. For each such set, we use 80% of the data as our training set and 20% as our test set. For learning the hierarchical model, we clustered grids based on past accident counts and the number of road intersections. A comparison of likelihoods is presented in Fig. III.4. We show that a hierarchical clustering approach improves (log)-likelihood significantly in all test sets, with an average improvement of about 13%.

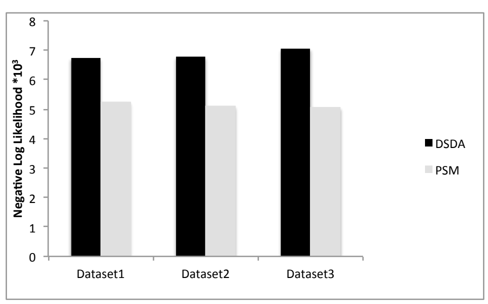


Figure III.4: Likelihood Comparison (Lower is better)

Next we evaluate the quality of severity prediction, using accuracy as a metric. The results are shown in Table III.1, where 3 severity classes were used. We find an average accuracy of about 66%, a reasonable performance on a 3-class classification problem. To delve more deeply, note that in emergency response settings, incorrectly predicting high severity incidents is more costly than overestimating severity. To evaluate this aspect, we also considered the fraction of times severity was underestimated (termed False Negatives) and overestimated (False Positives). We can see that the model rarely underestimates severity.

Table III.1: Severity Prediction Accuracy

Test Set	Percent Accuracy	False Negative	False Positive
Set 1	64.7%	3.4%	31.9%
Set 2	65.4%	1.6%	33%
Set 3	67.4%	1.8%	30.8%

6 Conclusion

We presented a novel discrete-space continuous-time model for forecasting incidents as a function of a collection of co-variates. Our model, which makes use of survival analysis, allows for spatial as well as temporal correlations with arbitrary features that affect incidents. An important contribution is also the ability of the model to capture the effect of police presence both temporally and spatially in order to predict crimes. For incidents who response is affected by their reported severity, we proposed learning a joint probability distribution over incident arrival and severity of incidents to tackle incident predictions heterogeneously based on priorities. We decomposed this distribution into a distribution over arrival times and a conditional distribution over incident severity given arrival times. To learn the former, we proposed a novel hierarchical clustering approach to extend the use of survival analysis to predict incidents by learning from data the spatial granularity of the model. We used a Multinomial Logistic Regression model to learn the distribution over incident severity.

For both crimes and traffic accidents, we validate our models using real crime and traffic accident data from Nashville, TN, USA. Specifically, for crimes, experiments demonstrate that our model outperforms state of the art continuous- and discrete-time crime prediction models both in terms of prediction effectiveness and running time. For accidents, we show that our Hierarchically clustered survival analysis model outperforms the standard survival analysis approach, thereby improving upon the model used to predict crimes. Further, and crucially, the survival model introduced in the chapter provides a structured and systematic way to use the incident prediction model as a simulator in making dispatch and placement decisions, which we look at in the upcoming chapters.

CHAPTER IV

Robustness in Incident Forecasting

1 Introduction

The increase in availability of data and algorithmic progress has created new ways of fighting crime, with predictive policing being on the forefront of such efforts. Such an approach to policing is primarily proactive, and involves using predictive analytics and data-driven methods to understand where crime could potentially happen. We described such an approach in Chapter III, that can predict events like crime in continuous-time and accommodate arbitrary risk covariates directly into the model.Developing models to predict crime is, however, only a part of the broader methodology of proactive policing. It complements the design of policies of intervention based on such models, such as deploying patrols to minimize expected response time to assist victims of crime. This involves creating principled methods of resource allocation based on prediction models, and then directing search and patrol resources accordingly. However, it is natural to assume that criminals gain knowledge of such strategies and adapt themselves, thus rendering predictive models stale and inaccurate. A major shortcoming of many prior predictive approaches is that they do not account for such adversarial adaptation.

This moivates us to create an algorithmic framework for predicting crimes that is robust to manipulations in criminal behavior. We seek to identify the vulnerabilities in crime prediction models, create a model for capturing adversarial actions and finally create a robust prediction model against such actions. We continue our work on Survival models in this chapter. Such models are generative, and can accommodate arbitrary sets of covariates, thereby making them especially suitable for learning spatial-temporal densities over incident occurrence. We restrict our focus to such models, and create an optimization-based framework for ensuring robust learning under adversarial manipulations. Specifically, we model the interaction between the learner and the attacker as a Stackelberg game, where the learner chooses a policing strategy and the attacker chooses to manipulate its behavior in response to the chosen strategy.

This game model between a learner and an attacker poses two major challenges: a) the resulting optimization problem is intrinsically difficult to solve due to the nested hierarchy of the attacker's and defender's optimization problems, and b) the set of attacker strategies is combinatorial. We explain how these specific challenges manifest themselves in our problem, and then devise two algorithmic approaches to solve it. Finally, we evaluate the performance of our approaches using real-world data from a major metropolitan city in USA. Our experimental results demonstrate that our approach is significantly more robust to adversarial manipulation than standard survival analysis models.

2 Model

We use the same setup as our original model in Chapter III, but describe it again to make this discussion self-contained. We consider a set of equally sized grids G, that spans the entire spatial area under consideration. Also, we consider a dataset D of crime incidents, in which incident $d_i \in D$ denotes the i^{th} incident. We assume that the spatial granularity for any learning model dealing with this area is captured by the grids. Consequently, it suffices to denote the spatial location of each incident by one of the grids. Further, each incident $d_i \in D$ is also characterized by a time of occurrence x_i and a set of spatial-temporal features $w_i \in \mathbb{R}^m$. Thus, our dataset is a collection of time-stamped feature vectors corresponding to crime incidents, $\{(x_1, w_1), (x_2, w_2), \ldots, (x_n, w_n)\}$. For each grid $g_i \in G$, we denote the time between successive incidents by the random variable τ, and define $t_i = x_i - x_{i-1}$ as the time to arrival of the ith incident in the dataset. Our primary goal is to robustly learn a distribution $f(t|w)$ (our notation of robustness will be defined below). We first describe a model to learn such a distribution, and then discuss the notion of robustness that we focus on.

2.1 Spatial-Temporal Prediction Model

Our goal is to learn a probability distribution over incident arrival in space and time. Given a probability distribution f for a continuous random variable τ (representing the inter-arrival time), a parametric survival model for a specific data point $\{\tau_i, w_i\}$ can be defined as

$$\log(\tau_i) = \theta_1 w_{i1} + \theta_2 w_{i2} + ... + \theta_m w_{im} + z, \tag{IV.1}$$

where, θ represents the regression coefficients and z is the error term, distributed according to the distribution h. The particular choice of the distribution f depends on how we model the error term z. We adopt a common exponential distribution model for τ, used previously in the context of incident prediction (Mukhopadhyay et al., 2016b). Using equation VIII.1, for a given collection of incidents D, the log-likelihood of the observed data is

$$L(D;\theta) = \sum_{i=1}^{n} \log h(\log(\tau_i) - w_i^T \theta). \tag{IV.2}$$

Given such a model for likelihood, the learner tries to find the parameters $\theta^* \in \Theta$, such that

$$\theta^* = \text{argmax}_{\theta \in \Theta} L(D;\theta). \tag{IV.3}$$

where Θ is the space of parameters.

2.2 Adversarial Intervention

The issue of robustness of survival models in the context of reliability generally evokes outliers, rather than deliberate manipulation. However, in an adversarial setting such as crime prediction, the use of survival modeling may expose its vulnerabilities, since criminals may deliberately shift criminal activity away from areas predicted as "*hot spots*" by the model, as this leads to increased police presence (Reppetto, 1976). Next, we describe the formal attack model, followed by a formal model of robust survival analysis in the context of our attack model.

2.2.1 Attacker Model

We assume that the attacker first observes the survival model f, and may shift to a different grid so as to commit a crime in an area with a smaller predicted crime frequency according to f. The reason for this shift in crime location is that predicted crime *hot spots*—that is, grids where crimes are predicted to be frequent—tend to elicit greater police presence. This, in turn, causes the predictions based on f to be inaccurate, and our ultimate goal will be to capture this effect in a principled way.

Consider a collection of potential incidents (for example, these can be incidents in our training data, as we discuss below), and the incident arrival model f. For each incident in the dataset, we assume that the attacker will react to f by potentially moving to a nearby grid (for some definition of "nearby", such as to an immediately adjacent grid). This assumption is based on two fundamental theories that govern crime occurrence: first, the opportunity theory of crime (Clarke, 2013; Hindelang et al., 1978), which posits that crime locations are deliberate choices by criminals driven by their attractiveness based on a specific utility function (which is often a combination of high reward and low risk); and prior crime literature (Chakravorty, 1995; Farrell and Pease, 2014), which dictates that similar crimes and opportunities are often clustered and occur close to each other. Therefore, even if there is an incentive for attackers to deviate from their ideal locations, they are

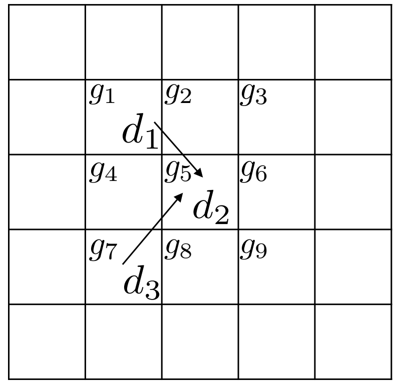


Figure IV.1: Adversarial Movement across grids

also averse to moving too far away, and may indeed be deterred in committing a crime if nearby opportunities do not present themselves.

To present the model formally, we now introduce some notation. For any grid $g_i \in G$, we use N_i to describe the set of neighboring grids that the attacker could move to in order to commit a crime (note that this includes the original grid, since the attacker could chose to not move). Further, for each grid $g_i \in G$, we use P_i to define possible successive (in time) pairs of data-points $\{d_k, d_l\}$ that could occur in g_i due to adversarial manipulation. We are specifically interested in successive incidents since the random variable we want to model is the inter-arrival time τ, which is the difference in time between consecutive incidents. We illustrate the idea behind such pairs in Figure IV.1. Consider the set of grids $\{g_1, .., g_9\}$ and incidents $\{d_1, d_2, d_3\}$, which we assume are ordered by their times of occurrences. Now, to look at adversarial perturbations in the dataset, we specifically look at grid g_5. Incidents in its neighborhood that could move to it form the set $\{d_1, d_2, d_3\}$ (note that this includes d_2 since the attacker could chose to not deviate from the original location of the incident). This gives us three pairs of successive incidents, namely the set $\{(d_1, d_2), (d_2, d_3), (d_1, d_3)\}$. Observe that (d_1, d_3) is also a *potential* pair of successive incidents in g_5 since the attacker could chose to move d_2 to a different grid, which would result in d_1 and d_3 occurring successively in g_5. Moreover, the pair (d_1, d_3) could exist as a pair of (possible) successive incidents if and only if d_2 moves to a different grid. In order to capture this, for any grid g_m and pair of incidents $(d_i, d_j) \in P_m$, we use B_{ij}^m to denote the set of all incidents $d_k \in D$ that could potentially move to g_m such that $t_i < t_k < t_j$. This lets us take into account the fact that d_i and d_j could occur consecutively in g_m if and only if they both move to g_m and none of the incidents from the set B_{ij}^m move to g_m. Finally, we capture the decision of the attacker to move an incident to a grid by the variable y_j^i, which is a binary variable that captures the decision of incident d_j shifting to grid g_i.

We are now ready to present the attacker's problem formally as the following optimization problem:

$$A(y;\theta) = \min_{y} \sum_{g \in G} \sum_{i,j \in P_g} y_i^g y_j^g \Big\{ \prod_{d_k \in B_{ij}^g} (1 - y_k^g) \Big\} f(t_{ij}, w_i, \theta) \tag{IV.4a}$$

$s.t.$

$$\sum_{g_i \in N_j} y_j^i = 1 \;\; \forall j \in \{1, .., n\} \tag{IV.4b}$$

$$y_j^i \in \{0,1\} \;\; \forall g_i \in G, \forall d_j \in D,$$

where the objective is the log-likelihood of incident arrivals *after* the shifts introduced by the attacker, with respect to the training data (which we use as a proxy for the distribution over future incidents).

In this optimization problem, the attacker considers all potential pairs of incidents that could occur in each grid due to possible manipulation, and chooses incidents such that the overall likelihood of the model is minimized. The product term in the objective function ensures that likelihood is captured only for decisions made by the attacker, and not for all the possible options that the attacker has. The attacker's action space in terms of which grids to move the crimes to is thus represented by the binary decision variables y, and constraint IV.4b ensures that the attacker can shift each incident to only one grid (since the same crime cannot be committed at the same time at two different locations).

2.2.2 Robust Survival Analysis

We model the problem of robust survival analysis as a Stackelberg game in which the defender first learns a model f, and the attacker then chooses an optimal shift in location in response, as described in the previous section. We assume that the defender's goal is to maximize the likelihood objective $A(y, \theta)$ over parameters θ, which the attacker aims to minimize by choosing crime location shifts y. Robust learning of the survival model can thus be represented as the following bi-level optimization problem:

$$
\begin{aligned}
&\max_{\theta} \min_{y} A(y,\theta) \\
&s.t. \\
&\sum_{i \in N_j} y_j^i = 1 \;\; \forall j \in \{1,..,n\} \\
&y_j^i \in \{0,1\} \;\; \forall g_i \in G, \forall d_j \in D,
\end{aligned}
\tag{IV.5}
$$

In the remainder of the paper we describe our approaches for solving this problem.

3 Approach

The optimization problem defined in equation IV.5 is not straight-forward to solve, due to the hierarchy of the *max* and *min* operators, as well as the combinatorial nature of the inner problem. In order to tackle this, we point out two key insights about the structure of the problem that we leverage to design a solution.

Insight 1 - We begin by taking a closer look at the attacker's objective over a set of binary decisions that represent choices of locations to commit crimes. The integrality constraints over y ensure the natural bound that the attacker can shift one incident to one particular grid. Now, consider relaxing the integrality constraints; this makes the attacker choose a probability distribution over grids to which it might shift its criminal activity. However, the integral solution still remains optimal, since the attacker's best response is a decision problem. Consequently, we can relax the integrality constraints without changing the quality of the solution.

Insight 2 - Secondly, we consider the primary consequence of the attacker's actions: choosing a location of crime effectively changes the inter-arrival times of incidents in grids such that the overall likelihood of the learned model is minimized. However, making such a decision is unrealistic from an attacker's perspective, since they would require the attacker to be effectively clairvoyant (they have to account for arrival time of incidents that haven't yet arrived). We therefore simplify the attack model; at any point in time t_i, we restrict the attacker to minimize the likelihood of the model for all incidents $d_k \in D$ such that $t_k \leq t_i$, since at this time, the attacker can only have information about incidents that have happened before t_i. This assumption dramatically reduces the complexity of the inner problem, since now the attacker's objective is reduced to an optimization problem over a finite set of grids - the attacker can shift each incident to the grid that results in lowest likelihood, without considering how such a decision can potentially affect future inter-arrival times.

These insights let us create two efficient approaches for solving the overall optimization problem that we present next. We begin by introducing our first approach, which is called **R**obust

Survival **A**nalysis with **L**inear **A**ttack (**RSALA**).

3.1 Robust Survival Analysis with Linear Attack (RSALA)

In order to address the optimization problem in formulation IV.5, we begin by reformulating the attacker's optimization problem as linear optimization with an exponential large number of variables. Combining this with constraint generation produces an algorithm that is guaranteed to converge to the optimal solution in finite time.

To begin, we note that each attacker action creates a specific set of shifts of crime incidents, resulting in a different modification to the original dataset. We refer to such a modified dataset as a *chain*. Thus, we can think of the attacker's full (combinatorial) action space as the set of all possible chains—that is, the set of all possible modifications of the data. Now, consider c to be the total number of such chains. The attacker's objective then reduces to choosing the chain that results in the lowest likelihood, which can be represented as:

$$
\begin{aligned}
&\min_{\lambda} \sum_{i=1}^{c} \lambda_i l_i(\theta) \\
&s.t. \\
&\sum_{i=1}^{c} \lambda_i = 1 \\
&\lambda_i \in \{0,1\} \;\; \forall i \in \{1,..,c\}
\end{aligned}
\tag{IV.6}
$$

where l_i represents the likelihood of the i^{th} chain.

The obvious issue with the optimization reformulation in Equation IV.6 is that c could be extremely large, making the problem intractable to solve. We aim to address this issue by looking at the dual of problem IV.6. First, we point out that the attacker can only choose multiple chains as part of an optimal solution if they contribute the same utility to the attacker's objective. This crucial insight lets us relax the integrality constraint over λ without sacrificing the utility of the attacker, and converts problem IV.6 into a linear program. Then, due to strong duality, we can directly replace the attacker's objective function in problem IV.5 with its dual, and represent the overall robust survival analysis problem as

$$
\begin{aligned}
&\max_{\theta,\delta} \delta \\
&s.t. \\
&\delta - l_i(\theta) \leq 0 \;\; \forall i \in \{1,..,c\} \\
&\delta \in \mathbb{R},\, \theta \in \mathbb{R}^m
\end{aligned}
\tag{IV.7}
$$

where $\delta \in \mathbb{R}$ represents the dual variable.

This formulation hands us two crucial advantages: first, it converts the *max-min* hierarchy of problem IV.5 into a single convex maximization problem, and secondly, it puts the potentially large number of possible attacker actions into a collection of constraints. This, in turn, allows us to solve the problem using constraint generation. A constraint generation approach starts with a subset of the attacker actions, and iteratively updates the model by dynamically generating constraints according to actions taken by the defender. We present this approach in Algorithm 2.

Algorithm 2 RSALA

1: **INPUT:** Dataset D, Likelihood Model L, Adversarial Utility function A
2: **OUTPUT:** Robust Survival Parameters θ^*
3: Set $\theta^0 \leftarrow \text{argmax}_{\theta\in\Theta}\, L(D;\theta)$; $k \leftarrow 0$; Constraint set $\phi^0 \leftarrow \text{Attack}(\theta^0)$; $gap \leftarrow \infty$
4: **while** $gap > \epsilon$ **do**
5: $\quad\theta^{k+1} \leftarrow \text{Solve}\ (\phi^k)$
6: $\quad\phi^{k+1} \leftarrow \phi^k \cup \text{Attack}(\theta^{k+1})$
7: $\quad D^{k+1} \leftarrow \text{Update}(D^k, \text{Attack}(\theta^{k+1}))$
8: $\quad gap \leftarrow L(D^{k+1}, \theta^{k+1}) - L(D^{k+1}, \theta^k)$
9: $\quad k \leftarrow k+1$
10: **end while**
11: **return** θ^{k+1}

We explain some added notation before explaining the algorithm. At any iteration k of the algorithm, we refer to the current set of constraints by ϕ^k, the defender's parameters by θ^k, and the dataset used in iteration k by D^k (which gets updated according to the actions taken by the attacker). Further, we use $Solve(\phi^k)$ to denote solving problem IV.7 under constraints ϕ^k, and use $Attack(\theta^i)$ to denote the generation of the attacker's best response against θ^i. Also, we use *Update(*D^k*,y)* to denote a function that updates the existing dataset with manipulations generated as a response to a specific choice of θ made by the defender (thereby arriving at a new dataset D^{k+1}). Now, at iteration k in the algorithm, we first compute the defender's optimal parameters θ^{k+1} by solving problem IV.7 under constraints ϕ^k (refer to step 5). We then update the constraint set by computing the attacker's response to θ^{k+1} (refer to step 6). Such a response is straight-forward to compute, due to insight 2 mentioned above. The attacker's response is then used to update the dataset (refer to step 7), which is then used in the subsequent iteration. This process is continued until the attacker's gain between successive iterations is within an exogenously specified parameter ϵ.

While *RSALA* is guaranteed to converge in finite time to the optimal solution, the strategy-space of the attacker could be extremely large, and solving the optimization problem IV.7 at every iteration of *RSALA* is computationally slow. This motivates us to create a heuristic approach, that can balance the trade-off between the quality of solutions and computation time of the algorithm. We call this approach **Ad**versary based **Grad**ient Descent (**AdGrad**).

3.2 Adversary Based Gradient Descent (AdGraD)

Traditionally, parametric survival models are optimized using gradient-based approaches. Our problem is not as straight-forward: attacker actions affect the model parameters, but these actions are a function of the model parameters as well. We modify the standard gradient-based approach to enable the defender to take gradient steps that are based on the attacker's adversarial actions, which intuitively, lets the defender adjust its learning to account for manipulations caused by the attacker. We present this approach formally in Algorithm 3. We use the same notation as in Algorithm 2, and denote the attacker's decisions at iteration k by $y(k)$. Given a dataset of prior incidents D, we initialize the defender's actions θ with the parameters of the standard survival model described in Equation VIII.2. Then, at each iteration of gradient descent, we first calculate the best response of the attacker (refer to step 5) under current parameters (θ) chosen by the defender. This provides us with an updated set of data with adversarial manipulations, which is then used by the defender to update its parameters using a standard gradient step (refer to step 7). This process is repeated until convergence, and we use the same notion of convergence as in *RSALA*.

Algorithm 3 AdGrad

1: **INPUT** Dataset D, Likelihood Model L, Adversarial Utility function A
2: **OUTPUT** Robust Survival Parameters θ^*
3: Set $\theta^0 \leftarrow \text{argmax}_{\theta \in \Theta} L(D;\theta)$; $k \leftarrow 0$; $gap \leftarrow \infty$
4: **while** $gap > \epsilon$ **do**
5: $\quad y(k+1) \leftarrow \text{Attack}(\theta^{k+1})$
6: $\quad D^{k+1} \leftarrow \text{Update}(D^k, y(k+1))$
7: $\quad \theta^{k+1} \leftarrow \theta^k + \alpha \nabla L(D^{k+1};\theta^k)$
8: $\quad gap \leftarrow L(D^{k+1}, \theta^{k+1}) - L(D^{k+1}, \theta^k)$
9: $\quad k \leftarrow k+1$
10: **end while**
11: **return** θ^{k+1}

4 Experimental Evaluation

4.1 Setup

We used crime data from a major metropolitan area of the US to evaluate our models. Specifically, we looked at data from two crime types: 7057 assault incidents and 2184 burglaries from 2014. We created three overlapping datasets for both types of crime, each having 4 months of data. For each dataset, we used 3 months of data as our training set and 1 month's data as our test set. For spatial discretization, we used square grids of side 1 mile throughout; this choice was a consequence of the fact that a similar granularity of discretization is followed by the local police department of the metropolitan area under consideration.

For the predictive model, we used prior work (Speer et al., 1998; Kennedy et al., 2011) and expert opinions (Private Communication, 2018) to choose features that can potentially affect the location of crimes. For each incident in a grid, we looked at prior crimes in the grid, prior crimes in neighboring grids, number of liquor stores and homeless shelters in the area, weather parameters like rain, temperature and snow, as well as socio-economic factors like mean household density and income.

The only hyper-parameters in our model are the learning rate α for *AdGrad* and the attacker's definition of "neighboring grids" to which the criminals can move. We set the learning rate using the second-derivative of the objective function, and define neighbors for a grid based on its adjacent grids. We also investigate the effect of varying the extent of shifts that the attacker has access to, but present the results later.

We ran all the experiments on a 2.4GHz hyperthreaded 32-core Ubuntu Linux machine with 32 GB RAM. While evaulating the algorithm *RSALA*, we solved the optimization problem (IV.7) using SciPy's non-linear optimization package (Jones et al., 01).

We use the model mentioned in Eq.VIII.2 as our baseline. This has two advantages - first, it allows us to evaluate the efficacy of our algorithms on survival models that are not explicitly trained to be robust, and secondly, it lets us compare our approach with a baseline that has shown better performance than other state-of-the-art alternatives in terms of accuracy of predicting crimes (Mukhopadhyay et al., 2016b). We refer to this model as *SSA*, which refers to standard survival analysis.

4.2 Results

To compare the efficacies of the two algorithmic approaches, we begin by looking at how the algorithms *AdGrad* and *RSALA* perform on unseen data, and directly compare their performance to

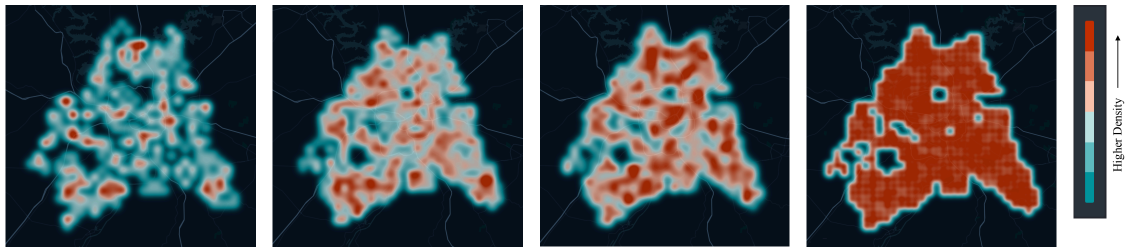


Figure IV.2: Predicted incident density for assault crimes plotted according to a varying attacker budget. Images from left to right are plotted with an attacker budget of 0, 1, 2 and 3 respectively.

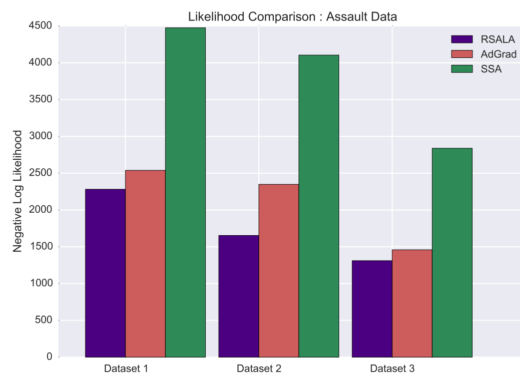


Figure IV.3: Test Set Likelihood (Lower is better) on Assault Data

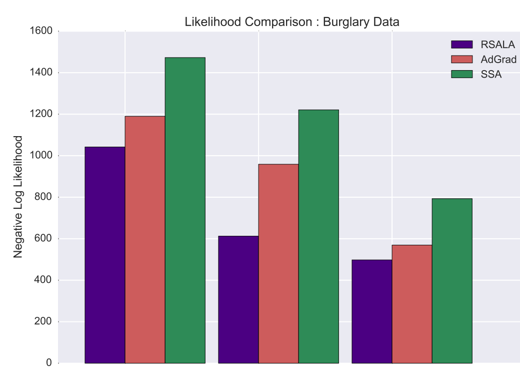


Figure IV.4: Test Set Likelihood (Lower is better) on Burglary Data

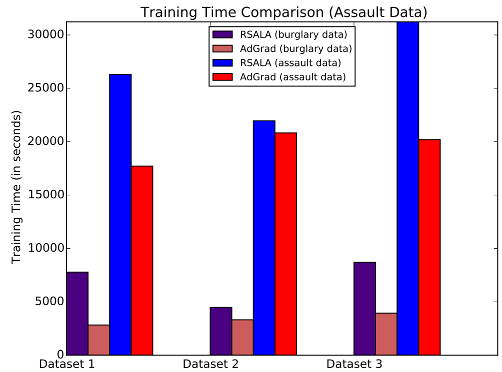


Figure IV.5: Training Time Comparison between RSALA and AdGrad

SSA. To do this, we introduce adversarial manipulations on our test data based on the attacker model described in section 2. We show the results on assault data and burglary data in Figures IV.3 and IV.4 respectively. In both cases, we observe that the both *RSALA* and *AdGrad* ensure significantly higher robustness against adversarial manipulations than *SSA*. Also, as expected, *RSALA* outperforms *AdGrad*, since it is guaranteed to converge to the optimal solution.

Next, we shed light on the time taken by the robust survival models to train, since training times for crime prediction algorithms can be a crucial factor in their deployment. Police intervention strategies are often computed periodically after shifts undertaken by police patrols, with the usual duration of such shifts varying from 6-8 hours (Amendola et al., 2011). Prediction models therefore, should have running times that do not affect deployment. We show our results for training times on burglary data and assault data in Fig IV.5. As expected, we see that *AdGrad* takes considerably less time than *RSALA* to train. On balance, we observe a clear tradeoff between the two algorithms: one allows us to compute an optimal solution, while the other, while somewhat suboptimal, can compute reasonablly good solutions quickly.

Finally, we seek to understand the effect of the attacker's geographic constraint (budget) on the robustness of the models. In order to do so, we vary the definition of "neighbors" that the attacker can shift to. We increase the attacker's budget gradually; consider a crime in a grid $g_i \in G$ and a budget of γ (say) - such a budget would enable the attacker to move to any grid $g_k \in G$, such that g_k and g_i have at-most γ other grids between them (a budget of $\gamma = 0$ reverts to *SSA*). With this notion of attacker's geographic budget, we repeat the entire set of experiments. Our findings for the performance of varying attacker budget are consistent with our findings shown in

Fig IV.3 and Fig IV.4. Instead, we seek to visualize the predictions made by the survival model as we increase the attacker's ability (i.e., as the criminals move farther away to commit crimes). Specifically, we plot the spatial-temporal survival density learned by varying the attacker's budget as heat-maps over the actual metropolitan area under consideration. We generate the heat maps by predicting assault crimes across all grids for 3 days, and we repeat this procedure 50 times to reduce variance in the predictions. We show the resulting images in Fig. IV.2, that are generated by attacker budgets between $0 - 3$. We see that as the attacker's budget increases, the survival models become increasingly cognizant about potential crimes occurring throughout the city, resulting in a spatial distribution of incidents that is spread out. An important insight revealed by this experiment is that a very high attacker budget can create models which essentially predict a high likelihood of crime occurring throughout the area under consideration, which is not necessarily useful in policing. Therefore, we point out that the attacker's budget is a crucial hyper-parameter in our models and recommend that system designers choose it carefully based on actual capabilities of the attacker, as well as opportunities that the spatial area under consideration provides for committing crimes.

5 Conclusion

Crime prediction models have traditionally been agnostic to adversarial manipulations in criminal behavior in response to learned models. We systematically bridge this gap by creating a principled nested optimization-based framework for predicting crimes that is robust to such manipulations. We specifically look at models based on survival analysis, and leverage the structure of the model to simplify the optimization problem, and propose two algorithmic approaches to solve the same. Finally, we use real-world data from Nashville to evaluate the efficacy of our approaches. Experimental results demonstrate that our approach is significantly more robust to adversarial manipulations than standard survival analysis.

CHAPTER V

Resource Allocation in Response to Crimes

1 Introduction

Prevention, response and investigation are the three major engagements of police. Ability to forecast and then effectively respond to crime is, therefore, the holy grail of policing. In order to ensure that crime incidents are effectively handled, it is imperative that police be placed in a manner that facilitates quick response. Effective police placement, however, needs crime prediction as a prerequisite. This is one of the reasons why predicting crime accurately is of utmost importance. Armed with a model that can predict crimes in time and space, we now look at how police can be optimally placed to respond to crimes. Before we introduce the structure of the problem, it is important to note how police patrols happen. Police vehicles operate in multiple ways - some vehicles patrol throughout the city performing the responsibility of preventing crimes (this effect however, is not well-established in literature). Other vehicles respond to crimes as and when they occur and return to their designated stations once the call is over. Usually, police vehicles operate in a combination of the two approaches. Our goal is to develop a rigorous optimization-based approach for optimal police placement in space in order to minimize expected time to respond to crime incidents as they occur. Throughout this chapter, we assume that a generative model for crime is available; and all references to a model to predict crimes refer to survival incident prediction models defined in Chapter III. While designing this decision theoretic problem, the key challenge we face is that crime locations and timing are uncertain. Moreover, for a given placement of police resources in space, optimizing crime incident response for a collection of known incidents is itself a non-trivial optimization problem. What makes this problem particularly challenging is that criminals are affected by police, as they avoid committing crimes if the chances of being caught are high; consequently, we expect that police placement will impact spatial and temporal distribution of crime incidents. Our model, therefore, has both decision and game theoretic features, even though we make use of a data-driven generative model of crime that accounts for the impact of police locations, rather than relying on rationality as underpinning criminal behavior.

Formally, we frame the problem of police patrol optimization as a regularized two-stage stochastic program. We show how the second-stage program (computing optimal response to a fixed set of crime incidents) can be formulated as a linear program, and develop a Bender's decomposition method with sample average approximation for the overall stochastic program. To address the fact that the top-level optimization decisions actually influence the probability distribution over scenarios for the second-level crime response optimization problem, we propose a novel iterative stochastic programming algorithm, *IBRO*, to compute approximate solutions to the resulting bi-level problem of finding optimal spatial locations for police patrols that minimize expected response time. We show that our model outperforms alternative policies, including the response policy in actual use by a US metropolitan police department, both in simulation and on actual crime data.

2 Optimizing Police Placement

Our goal is to address a fundamental decision theoretic question faced by police: how to allocate limited police patrols so as to minimize expected response time to occurring crime. In reality, this is a high-dimensional dynamic optimization problem under uncertainty. In order to make this tractable in support of practical decision making, we consider a simplified two-stage model: in the first stage, police determines spatial location of a set of patrol vehicles, P, and in the second stage, vehicles respond to crime incidents which occur. The decisions in the first stage are made under uncertainty about actual crime incidents, whereas for second-stage response decisions, we assume that this uncertainty is resolved. *A key strategic consideration in police placement is its impact on crime incidence.* In particular, it is well known that police presence has some deterrence effect on crime, which in spatio-temporal domains takes two forms: reduced overall crime frequency, and spatial crime shift Koper (1995); Short et al. (2008b). We assume below that the effect of police presence on crime distribution is captured in a stochastic crime model. Later, we describe and develop the stochastic crime model where we use real crime and police patrol data.

We present the problem formulation of allocating police given a stochastic generative model of crime. We divide the available area under police patrol into discrete grids. Formally, we define q as the vector of police patrol decisions, where q_i is the number of police vehicles place in grid i. Let s be a random variable corresponding to a batch of crime incidents occurring prior to the second stage. The two-stage optimization problem for police placement then has the following form:

$$\min_{q} \mathbb{E}_{s\sim f}[D(q;s)], \tag{V.1}$$

where $D(q;s)$ is the minimal total response time of police located according to q to crime incidents in realization s, which is distributed according to our crime distribution model f described in Chapter III, associated with each grid (and the corresponding spatial variables). The model implicitly assumes that crime occurrence is distributed i.i.d. for each grid cell, conditional on the feature vector, where the said feature vector captures the inter-dependence among grids. While the crime prediction model is continuous in time, we can fix a second-stage horizon to represent a single *time zone* (4-hour interval), and simply consider the distribution of the crime incidents in this interval.

The optimization problem in Equation (V.1) involves three major challenges. First, even for a given s, one needs to solve a non-trivial optimization problem of choosing which subset of vehicles to send in response to a collection of spatially dispersed crime incidents. Second, partly as a consequence of the first, computing the expectation exactly is intractable. Third, the probability distribution of future crime incidents, f, depends on police patrol locations q through the features that capture deterrence effects as well as spatial crime shift to avoid police. We address these problems in the following subsections.

2.1 Minimizing Response Time for a Fixed Set of Crime Incidents

While our goal is to minimize total response time (where the total is over the crime incidents), the information we have is only about spatial locations of crime and police in discretized space. As a result, we propose using distance traveled as a proxy. Specifically, if a police vehicle located at grid i is chosen to respond to an incident at grid j, the distance traveled is d_{ij}, distance between grids i and j. Assume that these distances d_{ij} are given for all pairs of grids i, j. Next, we assume that a single police vehicle is sufficient to respond to all crime incidents in a particular grid j. This is a reasonable assumption, since the number of crime incidents in a given cell over a 4-hour interval tends to be relatively small, and this interval is typically sufficient time to respond to all of them.

Given this set up, we now show how to formulate this response distance minimization problem as a linear integer program by mapping it to two classical optimization problems: the transportation Bertsimas and Tsitsiklis (1997) and *k-server* problems Chrobak et al. (1991).

In the transportation problem, there are m suppliers, each with supply s_i, n consumers, each with demand r_j, and transportation cost c_{ij} between supplier i and consumer j. The goal is to transport goods between suppliers and consumers to minimize total costs. To map crime response to transportation, let police vehicles be suppliers, crime incidents be consumers, and let transportation costs correspond to distances d_{ij} between police vehicle and crime incident grids, with each grid being treated as a node in the network. While the transportation problem offers an effective means to compute police response, it requires that the problem is balanced: supply must equal demand. If supply exceeds demand, a simple modification is to add a dummy sink node. However, if demand exceeds supply, the problem amounts to the multiple traveling salesman problem, and needs a different approach.

To address excess-demand settings, we convert the police response to a more general k-server problem. The k-server problem setting involves k servers in space and a sequence of m requests. In order to serve a request, a server must move from its location to the location of the request. The k-server problem can be reduced to the problem of finding minimum cost flow of maximum quantity in an acyclic network Chrobak et al. (1991). Let the servers be $s_1, ..., s_k$ and the requests be $r_1, ..., r_m$. A network containing $(2 + k + 2m)$ nodes is constructed. In the formulation described in Chrobak et al. (1991), each arc in the network has capacity one. The arc capacities are modified in our setting, as described later in the problem formulation. The total vertex set is $\{a, s_1, ..., s_k, r_1, ..., r_m, r_1^{'}, ..., r_m^{'}, t\}$. a and t are source and sink respectively. There is an arc of cost 0 from a to each of s_i. From each s_i, there is an arc of cost d_{ij} to each r_j, where d_{ij} is the actual distance between locations i and j. Also, there is an arc of cost 0 from each s_i to t. From each r_i, there is an arc of cost $-K$ to each $r_i^{'}$, where K is an extremely large real number. Furthermore, from each $r_i^{'}$, there is an arc of cost d_{ij} to each r_j where $i < j$ in the given sequence. In our setting, servers and requests correspond to grids with police and crime respectively. In the problem setting we describe, G is the set of all the nodes in the network. We term the set $\{s_i \; \forall i \in G\}$ as G^1, the set $\{r_i \; \forall i \in G\}$ as G^2 and the set $\{r_i^{'} \; \forall i \in G\}$ as G^3. The structure of the network is shown in Fig. V.1, which shows how the problem can be framed for a setting with 6 discrete locations. Shaded nodes represent the presence of police and crime in their respective layers.

The problem of finding placement of k-servers in space to serve an unordered set of requests is the same as the multiple traveling salesperson problem (mTSP), a generalization of the TSP problem, which is NP-hard. The offline k-server problem gets around this by having a pre-defined sequence of requests. By sampling crimes from the spatio-temporal model, although we can create a sequence of crimes by ordering them according to their times of occurrence, this sequence need not necessarily provide the least time to respond to all the crimes. In order to deal with this problem, we leverage the fact that crimes are relatively rare events. In order to find the ordering of crimes that provides the least response time, we solve the problem for each possible ordering of crimes. Despite this, the k-server solution approach is significantly less scalable than the transportation formulation. Consequently, we make use of it only in the (rare) instances when crime incidents exceed the number of available police.

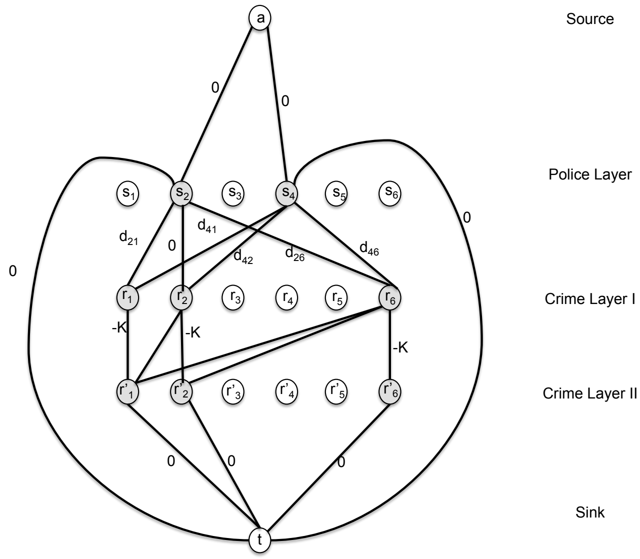


Figure V.1: Network Structure

2.2 Stochastic Programming and Sample Average Approximation for Police Placement

Now that we have two ways of addressing the problem of minimizing response time given a *known* set of crime incidents, we consider the original problem of optimizing allocation of police patrols. As a first step, we point out that the resulting stochastic program is intractable in our setting because of the large space of possible crime incident realizations. We therefore make use of sample average approximation, whereby we estimate the expectation using a collection of i.i.d. crime incident realization samples (henceforth, scenarios) generated according to f. For each scenario, we represent the presence of crimes in the grids by a binary vector z and total available police by k. The decision variable, x_{ij}^{s} refers to the number of police vehicles traveling from grid i to grid j in scenario s. Under such a setting, the optimization program with transportation problem in the second level can be formulated as:

$$\min_{q} \quad \sum_{s \in S} \left[\min_{x^s \geq 0} \sum_{ij} d_{ij} x_{ij}^s \right] \tag{V.2a}$$

$$\text{s.t.:} \quad q_i \in \mathbb{Z}_+ \ \forall i \in G$$

$$\sum_{i \in G} q_i = k \tag{V.2b}$$

$$\sum_{j \in G} x_{ij}^s = q_i, \ \forall i \in G, \forall s \in S \tag{V.2c}$$

$$\sum_{i \in G} x_{ij}^s = z_j^s, \ \forall j \in G, \forall s \in S, \tag{V.2d}$$

$$x_{ij}^s \geq 0 \ \forall i, j \in G \tag{V.2e}$$

The optimization program leveraging the k-server problem, on the other hand, can be formulated as:

$$\min_{q} \quad \sum_{s \in S} \left[\min_{x^s \geq 0} \sum_{ij} d_{ij} x_{ij}^s \right] \tag{V.3a}$$

$$\tag{V.3b}$$

$$\text{s.t.:} \quad q_i \in \mathbb{Z}_+ \ \forall i \in G$$

$$\sum_{j \in \{G^2, t\}} x_{ij}^s = q_i \quad \forall i \in G^1, \forall s \in S \tag{V.3c}$$

$$\sum_{i \in G} x_{ij}^s = z_j^s \quad \forall j \in G^2, \forall s \in S \tag{V.3d}$$

$$\sum_{j \in G} x_{ij}^s - \sum_{l \in G} x_{li}^s = s_i \quad \forall i \in G, \forall s \in S \text{ where } s_i = \begin{cases} k \text{ if } i = a \\ -k \text{ if } i = t \\ 0 \text{ otherwise} \end{cases} \tag{V.3e}$$

$$x_{ij}^s \leq 1 \quad \forall i, j \in \{i, j \in G\} \setminus \{\{i, j \in G \text{ and } i = a \text{ and } j \in G^1\} \cup \{i, j \in G \text{ and } i \in G^1 \text{ and } j = t\}\}, \forall s \in S \tag{V.3f}$$

$$x_{ij}^s \geq 0 \ \forall i, j \in G, \forall s \in S \tag{V.3g}$$

The overall optimization problem then becomes

$$\min_{q\geq 0}\mathbb{E}_{s\sim f}\left[\mathbb{1}(k\geq m_s)\min_{x^s\in C_1^{s(q)}}\sum_{ij}d_{ij}x_{ij}^s+\mathbb{1}(k<m_s)\min_{x^s\in C_2^{s(q)}}\sum_{ij}d_{ij}x_{ij}^s\right] \tag{V.4}$$

where $C_1^{s(q)}$ includes the Constraints V.2c, V.2d, and $C_2^{s(q)}$ includes Constraints V.3c V.3d and V.3e, as well as the capacity constraints, for all realizations of crime incidents s, that are drawn from the distribution f.

We propose to solve this stochastic program using Bender's decomposition Bertsimas and Tsitsiklis (1997). The first step is to represent the inner (lower-level) optimization problems using their duals, which for the transportation problem, is represented as:

$$\max_{\alpha,\beta}\sum_{i\in G}q_i\alpha_i^s+\sum_{j\in G}z_j^s\beta_j^s \tag{V.5a}$$

$$s.t.:\quad d_{ij}-\alpha_i^s-\beta_j^s\geq 0\quad\forall i,j\in G, \tag{V.5b}$$

where $\{\alpha_1^s,...,\alpha_g^s\}$ are the dual variables for Constraints V.2c and $\beta_1^s,...,\beta_g^1$ are dual variables for Constraints V.2d. The dual for the k-server problem is represented as:

$$\max_{\lambda,\delta,f,c}-\sum_{i\in G^1}\lambda_i^s q_i-\sum_{j\in G^2}\delta_j^s z_j-\sum_{i,j\in C_c}c_{ij}^s-\sum_{i\in G}f_i^s s_i \tag{V.6a}$$

$$s.t. \tag{V.6b}$$

$$\mathbb{1}(i,j\in C_\lambda)\lambda_i^s+\mathbb{1}(i,j\in C_\delta)\delta_j^s+f_i^s-f_j^s+\mathbb{1}(i,j\in C_c)c_{ij}^s+d_{ij}\geq 0\qquad\forall i,j\in G \tag{V.6c}$$

where

$$i,j\in C_\lambda\quad\text{if}\quad i,j\in G\quad\text{and}\quad i\in G^1,j\in\{G^2,t\}$$

$$i,j\in C_\delta\quad\text{if}\quad i,j\in G\quad\text{and}\quad i\in G^2$$

$$i,j\in C_c\quad\text{if}\quad i,j\in\{i,j\in G\}\setminus\{\{i,j\in G\text{ and }i=a\text{ and }j\in G^1\}$$

$$\cup\{i,j\in G\text{ and }i\in G^1\text{ and }j=t\}\}$$

We introduce dual variables $\lambda_i^s,...,\lambda_k^s$ for constraints V.3c, $\delta_i^s,...,\delta_m^s$ for constraints V.3d, $f_i^s,...,f_n^s$ for constraints V.3e and $c_{11}^s,c_{12}^s...,c_{nn}^s$ for constraints V.3f.

By construction, the primal transportation problem always has a feasible solution as it is balanced, and the primal k-server problem always has a feasible solution provided $\sum_i q_i > 0$, which is ensured by always having a budget greater than 0. Consequently, there always exists an optimal dual solution which is one of the (finite number of) extreme points of the polyhedron comprised from Constraints V.5b and V.6c for the corresponding problems. Since these constraints do not depend on the police patrol allocation decisions q, the set of extreme points of the constraint polyhedra $E^s = \{(\lambda^s, \delta^s, f^s, c^s)\}$ and $E^s = \{\alpha^s, \beta^s\}$ for both the problems are independent of q. Thus, we can then rewrite the stochastic program as

$$\begin{aligned} \min_q \quad \sum_{s \in S} \Bigg[& \mathbb{1}(k < m_s)\{ \max_{(\lambda^s, \delta^s, f^s, c^s) \in E^s} - \sum_{i \in G^1} \lambda_i^s q_i - \sum_{j \in G^2} \delta_j^s z_j - \sum_{i,j \in C_c} c_{ij}^s - \\ & \sum_{i \in G} f_i^s s_i \} + \mathbb{1}(k \geq m_s)\{ \max_{\alpha, \beta} \sum_{i \in G} q_i \alpha_i^s + \sum_{j \in G} z_j^s \beta_j^s \} \Bigg] \end{aligned} \tag{V.7}$$

Since E^s is finite, we can rewrite it as

$$\min_{q, u^s} \sum_s u^s \tag{V.8a}$$

$$\text{s.t.}: \quad q_i \in \mathbb{Z}_+ \quad \forall\, i \in G$$

$$u^s \geq - \sum_{i \in G^1} \lambda_i^s q_i - \sum_{j \in G^2} \delta_j^s z_j - \sum_{i,j \in C_c} c_{ij}^s - \sum_{i \in G} f_i^s s_i \quad \forall\, s, (\lambda^s, \delta^s, f^s, c^s) \in \tilde{E}^s \tag{V.8b}$$

$$u^s \geq \sum_{i \in G} q_i \alpha_i^s + \sum_{j \in G} z_j^s \beta_j^s \quad \forall\, s, (\alpha^s, \beta^s) \in \tilde{E}^s \tag{V.8c}$$

where $\tilde{E}^s$ is a subset of the extreme points which includes the optimal dual solution and Constraints V.8b and V.8c are applicable based on whether the particular scenario is mapped to the transportation problem or the k-server problem. Since this subset is initially unknown, Bender's decomposition involves an iterative algorithm starting with empty $\tilde{E}^s$, and iterating solutions to the problem with this subset of constraints (called the *master* problem), while generating and adding constraints to the master using the dual program for each s, until convergence (which is guaranteed since E^s is finite).

A problem remains with the above formulation: if police vehicles significantly outnumber crime events, we only need a few of the available resources to attain a global minimum, and the remaining vehicles are allocated arbitrarily. In practice, this is unsatisfactory, as there are numerous secondary objectives, such as overall crime deterrence, which motivate allocations of police which are geographically diverse. We incorporate these considerations informally into the following heuristic objectives:

- There should be more police coverage in areas that observe more crime, on average, and
- Police should be diversely distributed over the entire coverage area.

We incorporate these secondary objectives by modifying the objective function in (V.3) to be

$$\min_{q} -\gamma h_i q_i + \kappa q_i + \min_{x^s \geq 0} \sum_{s \in S} \sum_{ij} d_{ij} x_{ij}^s \tag{V.9}$$

where h_i is the observed frequency of crimes in grid i and γ and κ are parameters of our model. The first term $\gamma h_i q_i$ forces the model to place police in high crime grids. The second term κq_i penalizes the placement of too many police vehicles in a grid and thus forces the model to distribute police among grids.

2.3 Iterative Stochastic Programming

Bender's decomposition enables us to solve the stochastic program under the assumption that f is stationary. A key challenge identified above however, is that the distribution of future crime actually depends on the police placement policy q. Consequently, a solution to the stochastic program for a fixed set of samples s from a distribution f is only optimal if this distribution reflects the distribution of crime conditional on q, turning stochastic program into a fixed point problem. We propose to use an iterative algorithm, *IBRO (Iterative Bender's Response Optimization)* (Algorithm 4), to address this issue. Intuitively, the algorithm provides police repeated chances to react to crimes, while updating the distribution of crimes given current police positions. In the algorithm, MAX_ITER is an upper limit on the number of iterations, e is the set of all evidence (features) except police presence and $\tau|z$ is the response time to crime z. q and z, as before, refer to vectors of police placements and crime locations and $q_i|z_i$ refers to police placement given a particular set of crimes.

3 Results

3.1 Experiment Setup

We use the same experimental setup for the crime prediction algorithms as the one specified in chapter III. For solving the optimization problem described in Section 2, we use CPLEX version 12.51 . The experiments were run on a 2.4GHz hyperthreaded 8-core Ubuntu Linux machine with 16 GB RAM.

Algorithm 4 IBRO

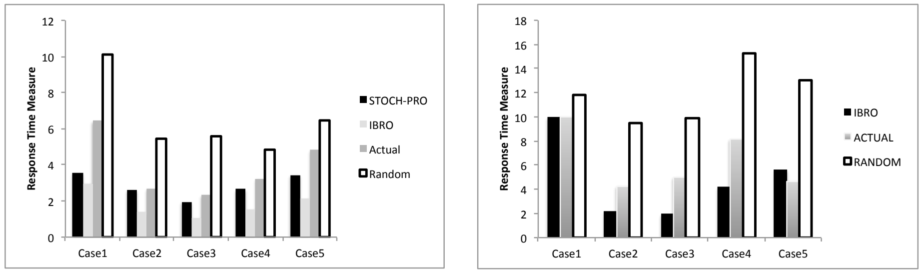
INPUT: q_0: Initial Police Placement
OUTPUT: q^*: Optimal Police Placement
for $i = 1..MAX\_ITER$ **do**
    Sample Crime z_i from $f(t|e, q_{i-1})$
    Find Optimal Police Placement $q_i|z_i$ by Stochastic Programming.
    Calculate $\mathbf{E_i}(\tau|z_i)$
    if $\mathbf{E_i}(\tau|z_i) > \mathbf{E_{i-1}}(\tau|z_{i-1})$ **then**
        Return q_{i-1}
    end if
    if $|\mathbf{E_i}(\tau|z_i) - \mathbf{E_{i-1}}(\tau|z_{i-1})| \leq \epsilon$ **then**
        Return q_i
    end if
end for
Return q_i

3.2 Effectiveness of the Response Time Optimization Method

Next, we evaluate the performance of our proposed framework combining iterative stochastic programming with sample average approximation. To do this, we randomly select timezones of 4 hours each from our dataset and sample 100 sets of crimes for each. In practice, although the number of police vehicles is significantly higher than the number of crimes in a 4-hour zone, all police vehicles are not available for responding to a specific type of crimes, due to assigned tasks. We consider a maximum of a single police vehicle per grid and we consider that only a fraction ($1/6th$) of the them are available to respond to burglaries. In order to simulate the actual crime response by the police department (in order to evaluate actual spatial allocation policy of police vehicles within our data), we greedily assign the closest police vehicle to a crime in consideration.

Our first evaluation uses our crime prediction model f to simulate crime incidents in simulation, which we use to both within the IBRO algorithm, as well as to evaluate (by using a distinct set of samples) the policy produced by our algorithm in comparison with three alternatives: a baseline stochastic programming method (using Bender's decomposition) which ignores the fact that distribution of crimes depends on the police allocation (*STOCH-PRO*), b) actual police location in the data (*Actual*), and c) randomly assigning police vehicles to grids (*Random*). Figure V.2(a) demonstrates that IBRO systematically outperforms these alternatives, usually by a significant margin.

Our next experiment evaluates performance of IBRO in comparison to others with respect to *actual crime incident data*. Note that this is inherently disadvantageous to IBRO in the sense that actual data is not adaptive to the police location as accounted for by IBRO. Nevertheless, Figure V.2(b) shows that IBRO typically yields better police patrol location policies than either actual (in the data) or random.



(a)

(b)

Figure V.2: Response Times (lower is better): (a) using simulated crimes, (b) observed crimes.

4 Conclusion

We develop a novel bi-level optimization method for allocating police patrols in order to minimize expected crime incident response time. Our approach makes use of stochastic programming, with a Bender's decomposition and constraint generation framework offering a scalable solution approach. Moreover, we introduce a novel iterative stochastic programming algorithm which allows us to account for the dependence of the spatio-temporal crime incidence distribution on police location. To evaluate this optimization framework, we presented a novel discrete-space continuous-time model for forecasting crime as a function of a collection of co-variates which include vehicular police deployment.

CHAPTER VI

Resource Allocation in Response to Time-Critical Emergencies

1 Introduction

A category of emergencies involve incidents where the most crucial factor for response is the time taken to respond. Incidents like fire, accidents, distress calls are some of the examples that fall under this category. It is essential to respond as fast as possible since these incidents often involve loss of life and property. From the perspective of a emergency responder organizations, two problems are pivotal: 1) how to deploy limited resources, such as fire depots and vehicles, so as to best anticipate potential future incidents, and 2) how to respond to emergencies as they occur within *specified time limits*. We focus on the second problem here since it is must be dealt with before actually responding to incidents. Emergency responder resources are stationed in different kinds of depots or stations (we use these words synonymously through the rest of this text). These stations are of various types, ranging from permanent structures to rented parking places. This makes the optimization over depot locations tricky as it is often infeasible for organizations to move some of the stations. However, governments periodically evaluate the location of such stations and relocate if needed, which motivates us to look at this problem. Moreover, the fire department of Nashville has several stations that are rented spaces which are relatively easy to relocate. Another important decision-making problem is the distribution of incidents in such depots, which from an administrative as well as practical standpoint, is an easier relocation process. What makes these problems difficult that the dependence on not only on how, when and where incidents happen, but also on the severity of the incidents. Moreover, such decisions are often times extremely critical as a matter of seconds can be life-saving. In fact, the American Planning Association states that the strategic location of fire-stations can prove to be the difference between lives and deaths, and salvage and destruction. It also mentions that in practice, a single fire station that is strategically located can perform better than several stations that are sub-optimally located. These concerns make it extremely important to design principled approaches to deal with this problem.

In this chapter, we systematically address the problem of finding optimal location of responder stations and distribution of responders in these, the latter being the key distinction between this and the well-known facility location problem (Guha and Khuller, 1999; Li, 2011). We develop a novel optimization problem to maximize incident density coverage with restrictions on waiting times and considering incident priorities, drawing on results from queuing theory. Our approach builds on the optimization method by Silva and Serra (Silva and Serra, 2008), but they restrict the approach to a single responder per station, a major limitation in practice. Our extension entails a non-trivial technical contribution and makes the approach far more practically viable. We note that while most prior work deals with response optimization and incident prediction separately, the latter is a fundamental requirement of the former. Thus, in order to aid our optimization model and validate it, we use the hierarchically structured probabilistic survival model described in Chapter 2. Since the severity of incidents is crucial in making allocation decisions regarding responders, it is extremely beneficial that our incident prediction model can also predict the severity of incidents. We perform extensive experiments using real traffic data from the metropolitan area of Nashville and show that our principled approach outperforms existing allocating strategies used in practice.

We evaluate our proposed approach using traffic accident data obtained from the fire department of Nashville, US (perhaps counter-intuitively, traffic accidents comprise the most common incident type to which this fire department responds). We demonstrate that our approach is a significant improvement over the state of the art in terms of incident prediction efficacy, and our method for locating responder stations substantially improves upon the current approach actually in use in this US city.

.

2 Optimizing Responder Placement

A fundamental problem faced by emergency responding agencies is to optimally allocate depots in space, allocate response vehicles in depots, and assign vehicles to incidents. As mentioned before, depots commonly refer to fixed responder stations, such as police and fire stations, which are predetermined. In other settings, depots can be periodically reallocated. Nevertheless, the time scales of depot and response vehicle allocations are typically different. Our approach can be modified directly if we wish to reassign vehicles to a fixed set of depots at an hourly or daily time scale, with the full problem (including depot allocation) solved at longer (say, monthly) time scales.

While the notion of optimality in emergency response problems can vary, we focus on the problem of maximizing coverage, one of the most common variants in this domain. To start, we discretize space into areas (henceforth referred as a set of grids G). Suppose that for each grid g_i, incidents of severity k arrive at a (predicted) rate λ_i^k. We assume that a predictive model for the concerned incident type is available. Given the arrival rates, for each grid g_i and incident severity k, jointly referred to as a pair (i,k) henceforth, we aim to allocate a depot d to respond to the associated incidents. We refer to a successful assignment of (i,k) to a responder depot d as *covering* the pair. We note that a measure of importance of such a pair is its predicted arrival rate λ_i^k. We try to maximize the total arrival rate that the model covers by optimally placing depots in a subset of the available grids, where each depot can hold a collection of responder vehicles. Thus, given a set G of discrete grid locations, p different responders (emergency vehicles) and a budget to allocate b different depots, we want to find the optimal location of the depots and the distribution of vehicles in such depots.

We now describe the formal structure of this optimization problem. For simplicity, we index depots by their grid numbers, which means that a depot located in grid j is referred to as depot j. Moreover, when there is no responder available in a depot to serve an incident that is assigned to it, we assume that the incident enters a waiting queue. We assume that each depot maintains its own queue which is ordered according to incident priorities but is non-preemptive at service time, which means that an incident already getting responded to is never left midway to attend to an incident of higher priority. In our model, lower values of k correspond to higher priorities. A similar approach has been studied previously with the aim of maximizing the total population covered (Silva and Serra, 2008). However, we look at a generalized problem structure where more than one responder can be placed at a location, which significantly complicates the queuing model in consideration by changing a single-responder priority queue model to a multi-responder priority queue model.

Formally, we consider the following optimization problem.

$$\max_{x,y,d} Z = \sum_k \sum_j \sum_i \lambda_i^k x_{ij}^k \tag{VI.1a}$$

$$s.t.: x_{ij}^k \leq d_j \quad \forall i,j \in I, \forall k \tag{VI.1b}$$

$$x_{ij}^k \leq y_j \quad \forall i,j \in I, \forall k \tag{VI.1c}$$

$$\sum_{j \in I} x_{ij}^k \leq 1 \quad \forall i \in I, \forall k \tag{VI.1d}$$

$$y_j \leq y_j d_j \quad \forall j \in I \tag{VI.1e}$$

$$\sum_{j \in I} y_j \leq p \tag{VI.1f}$$

$$\sum_{j \in I} d_j \leq b \tag{VI.1g}$$

$$w_j^k \leq \tau^k \quad \forall j, \forall k \tag{VI.1h}$$

$$y_j \in [1..p] \quad \forall j \in I \tag{VI.1i}$$

$$x_j, d_j \in \{0,1\} \quad \forall j \in I \tag{VI.1j}$$

where I is the set that indexes over all grid numbers, d_j is a binary decision variable which is 1 if there is a depot located at grid g_j, y_j is a decision variable that indicates how many responders are placed at depot j and x_{ij}^k is a binary decision variable which is 1 if depot j is assigned to respond to the pair (i,k) and 0 otherwise. We ensure that service standards are met by enforcing constraints that the mean waiting time (denoted by w_j^k where j and k correspond to the depot number and the priority respectively) at all depots is less than a pre-specified time limit τ^k. We assume that this information is user-specified, depending on the type of incident and the service quality required. The objective (VI.1a) aims to maximize the total coverage by the responders. Constraint (VI.1b) ensures that calls are assigned to locations that have depots, constraint (VI.1c) forces that such depots have at least one responder assigned and constraint (VI.1e) ensures that responders are placed in locations which are depots. Further, constraint (VI.1d) ensures that each pair (i,k) is assigned at most once. Constraints (VI.1f) and (VI.1g) are budget restrictions on responders and depots respectively and finally, constraint (VI.1h) ensures that the mean waiting time for incidents is within a pre-specified tolerance.

Before we attempt to solve this problem, we present a method to calculate the waiting time for a given depot. We assume that arrival rates for incidents are available; such rates can be learned from an incident arrival model like the one described in Chapter III. We model the inter-arrival time of incidents as exponential, and consequently the arrivals are Poisson distributed. We make the standard assumption that the service times are exponential as well, giving us a queuing model with memoryless arrivals, memoryless service times and multiple servers, commonly represented as a $m/m/c$ priority-queue model using the Kendall's notation (Kendall, 1953). Such a model is difficult to analyze when multiple priority events are present and each follows its own service time distribution (Harchol-Balter et al., 2005). We make a simplifying assumption that although different severities follow different arrival distributions and have different service time constraints in the optimization model, they follow the same service distribution. Thus, in our formulation, we assume that all priorities are served with a common exponential distribution with mean μ. This is an assumption that is realistic in many real-life applications as severities often represent the urgency with which an incident needs to be responded to and is not an indicator of actual service time. Moreover, analysis on our dataset revealed that learning the same distribution across event severities appears to be nearly as good as learning heterogeneous distributions (see Section 3, Table VI.1). We present a sketch of the derivation for the waiting time of our queuing model here. The full derivation can be found in prior literature in queuing theory (Cobham, 1954).

2.1 Calculating Waiting Time

Consider that an incident of priority k happens and has to enter the waiting queue for depot j with y_j responders because n_0 incidents of higher or equal priority are already waiting to be serviced. Also, let Λ denote aggregate arrival rates, such that $\Lambda_j^k = \sum_{i \in I} x_{ij}^k \lambda_i^k$,and let $\Lambda_j = \sum_k \Lambda_j^k$. Λ_j^k thus measures the rate of arrival of incidents of priority k for all grids that depot j serves. Let us assume that it takes t_0 time to service n_0 incidents. However, in time t_0, all arrivals of higher priority will supersede our event in the queue. Let there be n_1 such events which can be served in time t_1. Again, there can be further arrivals in t_1, and so on. Our incident, therefore, must wait for time $\sum_{l=1}^{\infty} t_l$ before it is serviced. Since we want the expected waiting time, we want to calculate $\mathbb{E}(\sum_{l=1}^{\infty} t_l)$. This can be calculated by looking at the conditional waiting time $\mathbb{E}(t_{l+1}|t_l)$, which is given by $\frac{1}{y_j \mu} \sum_{q=1}^{k-1} \Lambda_j^q t_l$, where μ is the mean service time distribution. Now, for any h, $\mathbb{E}(\sum_{l=0}^{h+1} t_l)$ is given by $\mathbb{E}(\sum_{l=0}^{h} t_l + \mathbb{E}(t_{h+1}|t_h))$. By induction and considering $h+1 \rightarrow \infty$, we get an expression for the average waiting time for an incident with priority k as

$$w_j^k = \frac{\frac{\pi}{y_j \mu}}{(1 - \frac{1}{y_j \mu} \sum_{q=1}^{k-1} \Lambda_j^q)(1 - \frac{1}{y_j \mu} \sum_{q=1}^{k} \Lambda_j^q)}$$

where

$$\pi = \frac{\left(\frac{\Lambda_j}{\mu}\right)^{y_j}}{y_j!\left(1-\frac{\Lambda_j}{y_j\mu}\right)\left[\sum_{r=0}^{y_j-1}\frac{\left(\frac{\Lambda_j}{\mu}\right)^r}{r!}+\sum_{r=y_j}^{\infty}\frac{\left(\frac{\Lambda_j}{\mu}\right)^r}{y_j!y_j^{r-y_j}}\right]}$$

We note that the queuing model assumes that the service time distribution is memoryless. This is a concern as the time taken by a responder to travel to an incident is not distributed exponentially. To tackle this, we assume that a depot can only respond to an incident if it is located within a small distance s of the incident, which in practice is sufficiently small that it can be treated as constant with respect to the overall service time.

2.2 Adaptive Random Search for Responder Optimization

The main challenge in solving mathematical program (VI.1) is the fact that Constraints (VI.1h) are non-linear and non-convex. We tackle this problem using greedy random adaptive search (GRASP). Such a procedure has been previously used in coverage maximization (Silva and Serra, 2008), but this previous approach cannot be directly applied when depots can have multiple responders as the search space becomes significantly more complex. We therefore propose a novel algorithm, **H**euristic based **R**esponse **O**ptimization of **C**overage with **Q**ueuing (*HROCQ*). We break up the algorithm into two parts and describe the construction of the Restricted Candidate List in Algorithm 5 and the Local Search Phase in Algorithm 6 in sequence.

We first describe the construction of the Restricted Candidate List (*RCL*; Algorithm5). We use D_j to denote the set of all nodes within a distance s of node j and D_{ij} to denote the i^{th} element of this set. We use $\bar{j}$ to denote all grids that have never been assigned a responder in the course of our iterative algorithm, $\text{Distribute}_b^i(a, h)$ as a method to distribute responders from grid a to i grids in set h in proportion to their demand rates (absence of i means that the distribution is done to as many nodes as possible) with a limit of b on the total number of grids that have responders, and NoDemands(a) as a method that returns *True* if no valid assignment can be made to node a and returns *False* otherwise. We use $|S|$ to denote the number of responders in our solution set and $||S||$ to denote the number of grids in S, that we iteratively build. Also, consistent with the optimization problem formulation, at any point in the algorithm, the number of responders assigned to grid j is denoted by y_j.

In order to decide the number of servers to be placed in a depot, we first sort the depots according to their event demand rate. We refer to this sorted node list as SN. Then, for the g^{th} run of the construction phase, we greedily assign p responders (our budget) to the first g depot locations

Algorithm 5 Restricted Candidate List Construction

INPUT: λ; p, K
OUTPUT: RCL : Restricted Candidate List
Initialize $S : \phi$
Create sorted list SN of candidate sites with respect to population/demand rate.
for $g = 1$ to b **do** ,
    Assign p servers to g grids according to λ.
    while $|S| \neq p$ **do**
        for $j \in SN$ **do**
            for $k = 1$ to K **do**
                if NoDemands(j) **then**
                    Distribute$_b(j, \bar{j})$
                else
                    for $i \in D_j$ **do**
                        if $w_j^k < \tau^k$ **then**
                            Set $x_{ij}^{[k]} = 1$
                        end if
                    end for
                end if
                Calculate $\Lambda_j^k = \sum_{i \in D_j} x_{ij}^k \lambda_i^k$,
            end for
        end for
        Construct RCL
        select $j^* =$ RandomSelect(RCL)
        Update $S := S \cup j^*$
        Remove demand nodes assigned to j^*
    end while
    Calculate Z_g
end for
Return Allocation S with highest Z.

in the SN, in proportion to their demand rates. Then, iteratively, for each node j that has been assigned a responder, we inspect D_j. For each node i in D_j and priority k (starting from the highest priority), we assign depot j to respond to the pair (i, k). After making each assignment, we ensure that no waiting time constraint is violated. We stop assigning calls to a depot when its waiting time constraint is violated and move to the next depot location in SN. After a phase of assignments, we look at the total demand rate Λ_j. We identify the highest serving depot as $j = \text{argmax}_i \Lambda_i$ and its corresponding service rate as $\Lambda_j^{'}$. Then, we select all nodes in our RCL that have a service rate of at least $\gamma \Lambda_j^{'}$. Finally, to finish one run of an assignment, we randomly select a node from the RCL and permanently fix its assignments and remove the pairs assigned to it from being considered in the future.

We stop when all depots that were assigned responders have been assigned a pair of calls, or when there are no more pairs to assign. This entire process is run b times, and we get a feasible solution from each such run, which we then carry forward to the local search phase, described next.

In the local search phase, described in Algorithm 6, we iteratively look at each depot from the current solution and deallocate all pairs assigned to it. We also deallocate the responders that were assigned to this depot. We distribute these responders iteratively to other potential depots not in the current solution set in proportion to their demand rates. We then calculate the updated objective value by replacing the unassigned node with the newly assigned set of nodes (referred to as S^i), where i is the number of grids that have received the freed responders. Finally, if any such assignment improves the objective value, we accept the updated solution. This method, repeated iteratively, performs a local search both with respect to the depots and the number of servers per depot.

Algorithm 6 Local Search Phase

INPUT: Restricted Candidate List
OUTPUT: Updated Solution
$Z^* := Z^S$, objective with current solution set S
for j in S **do**
    $\bar{S} := G \setminus S$ Find all nodes that are not in the Solution
    $S := S \setminus j$ Deallocate assignments
    Deallocate y_j responders
    for $i = 1 \text{ to } ||\bar{S}||$ **do**
        $\text{Distribute}_b^i(j, \bar{S})$
        Calculate Z^{S^i}
    end for
    $i^* := \text{argmax}_i Z^{S^i}$
    if $Z^{S^{i^*}} > Z^*$ **then**
        $S = S^{i^*}$
        $Z^* = Z^{S^{i^*}}$
    else
        Revert Deallocations.
    end if
end for
Return S

3 Results

We present the results in two different categories. First, we evaluate the effectiveness of our approach in predicting incidents, and second, we evaluate the performance of our responder optimization methods.

3.1 Data

Our evaluation uses traffic accident data obtained from the *fire department* in Nashville, USA, with a population of approximately 700,000. For this fire department, traffic accidents comprise a large majority of incidents it responds to (fires, in contrast, are relatively rare). We looked at data for 26 months, from 2014 - 2016, comprising of a total of 20148 traffic accidents. Each accident is accompanied by its time of occurrence, the time at which the first responding vehicle reached the scene and the time at which the last responding vehicle was back at service, which refers to completion of servicing an incident. To predict incidents, we extracted highway and street intersections from Open Street Maps (Haklay and Weber, 2008) and weather data was collected at the county level.

Before presenting the results, we highlight the three central problems faced by the fire department that are addressed by our framework: 1) optimal choice of locations for the fire depots, 2) optimal decision about which vehicles should reside in which depots, and 3) optimal decision about which depots are assigned to respond to which traffic accidents, with particular emphasis on minimizing cross-depot dispatch due to considerations such as vehicle-maintenance. The third of these is a particularly acute concern, as they view their current policy of assigning vehicles to accidents to be inefficient, unnecessarily increasing response times as well as mileage on the fire vehicle fleet (the latter resulting in more frequent and costly repairs).

3.2 Incident Prediction

Note that the performance of both the arrival prediction methodology as well as severity prediction can also be measured indirectly by the performance of the response optimization framework that uses predicted data to learn arrival rates. We address this next.

3.3 Response Optimization

Having looked at the performance of our predictive model, we now evaluate the performance of the response optimization model. To do this, we show several measures of performance of our model in a way that highlights its performance with respect to actual patrol policies and sheds some light on the structure and the components of the model. To achieve this, we use the same data sets for validating the response optimization mechanism as the training mechanism. First, we show how the components of *HROCQ* aid its performance. The two main stages of the algorithm involve *a)* greedily building a restricted candidate set for each and then *b)* choosing a candidate depot location from the set to form a partial solution to the problem. We evaluate the importance of the local search phase that works over the greedy-random candidate list phase. We present results by varying the number of responders and the parameter γ and check how the two stages of GRASP work. The results are presented in Fig. VI.1. For different problem configurations, we show how the objective function varies between the two phases. We note that the local search phase usually improves the existing solution. Although the improvement is typically small, it is crucial in emergency responder placement.

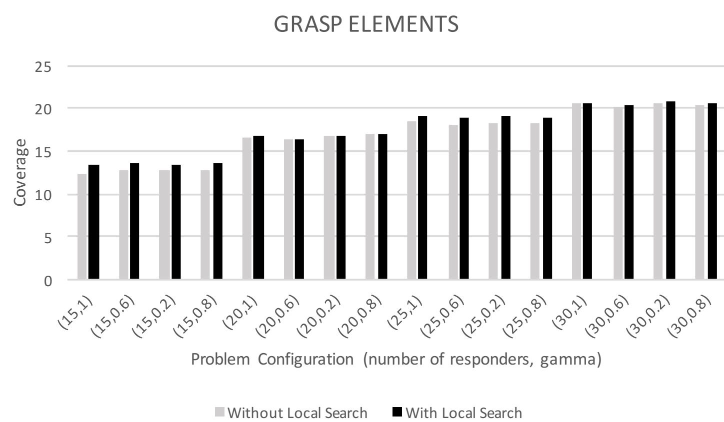


Figure VI.1: Phases of GRASP: Objective Value Comparison

Having seen the importance of the local search phase, we turn to tuning the parameter γ in our model. For a run of the algorithm, γ is fixed *a priori*. The parameter can be interpreted by understanding that $\gamma = 1$ corresponds to a completely greedy construction and $\gamma = 0$ corresponds to a construction that recognizes all temporary depot locations as candidate solutions in the *RCL* phase. In order to determine the parameter, we perform a discrete search over its domain, by varying it from 0 to 1 in steps of 0.1. We looked at cumulative results over 20 runs to determine the average objective value attained by each value of γ. We present the results for one training set in Fig. VI.2. Observe that usually a γ of 0.7 produces the best placement of responders and depots.

Finally, we compare our model to the existing responder deployment strategy in Nashville, TN. The accident data that we have reveals 3 types of priorities among incidents, namely, *A*,*B* and *D*, in order of increasing priority. First, we validate our assumption that learning a common service time distribution across event priorities is sufficiently accurate. We compare the *AIC* scores of two

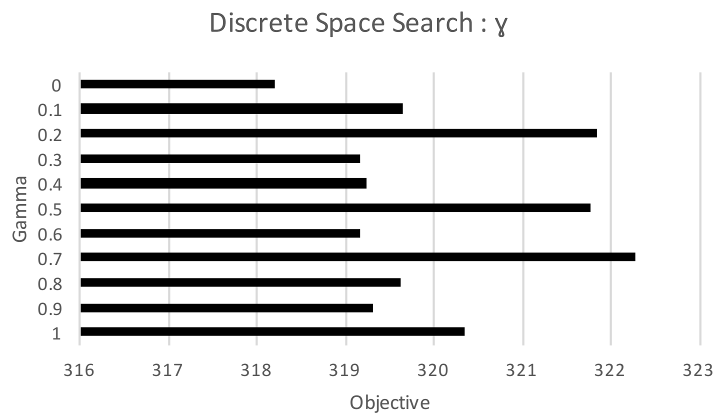


Figure VI.2: Discrete Search over γ's Space

models: *a)* Learning a separate service time distribution for each of the priorities and *b)* Learning a common distribution across event priorities. The AIC Score is defined as: $\text{AIC}_m = 2k - 2\ln(L)$ where k is the total number of parameters estimated and L is the likelihood of the data under model m. We present the findings in Table VI.1, which shows that there is little loss in assuming a common service time distribution across event priorities.

Table VI.1: AIC Score Comparison : Service Distribution Models

Model	AIC Score
Separate Servicing Models for each Priority	326724.55
Common Servicing Model	326721.16

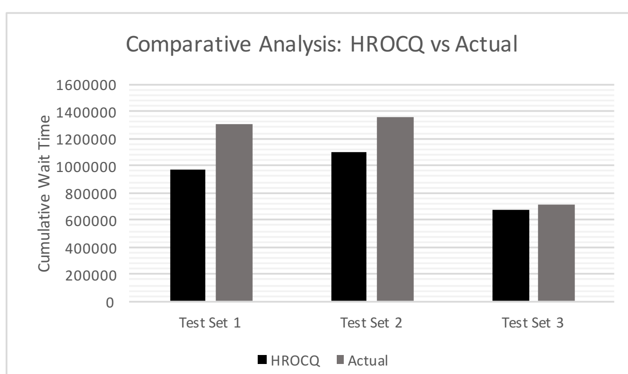


Figure VI.3: Waiting Time Comparison : *HROCQ* vs Actual

Next, we evaluate the overall performance of *HROCQ*. To compare the waiting time to serve incidents, we calculated the performance of *HROCQ* on our traffic accident data. The total number

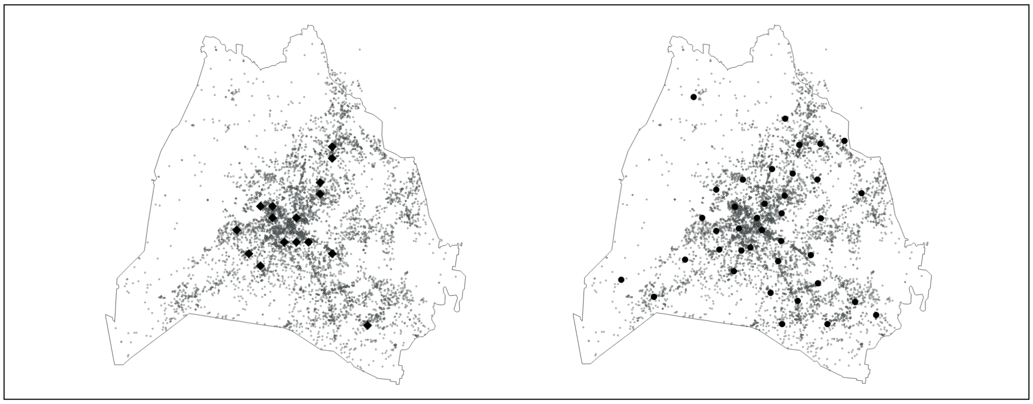


Figure VI.4: Location of Fire Stations: Optimized (Left) and Actual (Right)

of responders in the model is taken as 25, which is the number of fire department vehicles available. We fix the waiting time upper bounds as 4 minutes, 8 minutes and 12 minutes for categories *D, B and A* respectively. Also, we set the value of s (max travel distance) as 3 miles, whose travel time is small with respect to service times (an assumption made in the model). Before presenting the results, we point out that in our dataset, each incident is marked with the time when the first responder arrives at the scene as well as the time when the last responder returns after servicing an incident. We assume that these are the exact emergency responders needed, which is not always the case; consequently, it offers a *conservative* evaluation of the relative performance of our approach compared to actual response times. As an example, a common scenario for an accident is that the nearest police vehicle visits it first and the actual medical response vehicle arrives later. Similarly, while emergency responders return when their task is finished in the real world scenario, our model's validation must assume (due to unavailability of data) that the last vehicle to get back to service was the medical response vehicle, increasing wait times.

The mean travel time to incidents in our dataset is about 2.14 minutes. Although this is small in comparison to the service time, and is not explicitly contained in the model, we take into account the travel time in our validation to draw a fair comparison to the existing strategy used by the Fire Department. For each test dataset, we make a comparison based on total waiting time across all incidents and present the results in Fig. VI.3. We note that in all cases, the proposed formulation reduces the total service time, with an average improvement of about 16%, with the comparison heavily weighted against our approach. We highlight that this is a remarkable improvement, both in terms of performance and due to the importance of the incidents served, which often involve life-and-death scenarios. We show in Fig. VI.4 the difference in placement of actual responder placement stations and the station placed by our model. We observe that *HROCQ* focuses on areas with

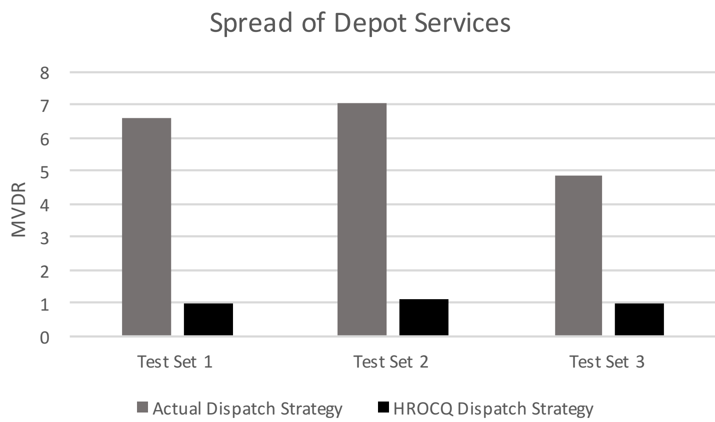


Figure VI.5: Spread of depots responding to a location

high density of incidents by placing more than one responder in these depots rather than distributing responders across the urban area. The low density areas are then covered by stations in high density areas as and when required.

It was observed in the urban area of concern that the incidents happening in the same location are serviced by responders from many distinct depots, which is likely a consequence of greedily assigning the closest responder to each incident. This naive approach makes responders from depots with high densities extremely busy and unavailable when accidents occur in their designated areas, and is viewed as a major concern by the fire department. We now consider how *HROCQ* addresses this issue compared to the actual responder policy. We point out the *HROCQ* implicitly tackles this problem by assigning grids to specific depots. However, not all potential demand locations are covered, and in such scenarios, other depots are called into action. To compare the results, we calculate for each grid i the average number of depots that respond to it; we denote this quantity by r^i. We show how structured the dispatch mechanism is by calculating $(\sum_i r_i)/(\sum_i i)$, which we refer to as the mean variation in depot response ($MVDR$). We see how $MVDR$ varies in practice versus *HROCQ* in Fig. VI.5. We can observe that the proposed approach provides a far more structured response mechanism.

4 Conclusion

We proposed a novel optimization problem that maximizes coverage of locations that need response while maintaining service time requirements by finding optimal location of response depots as well as the distribution of responders in them. In order to solve the optimization problem, we then extended prior work by proposing a novel greedy random adaptive algorithm that can accommodate the presence of multiple responders per demand node.

In order to predict incidents to aid the optimization model, we proposed learning a joint probability distribution over incident arrival and severity of incidents to tackle incident predictions heterogeneously based on priorities. We decomposed this distribution into a distribution over arrival times and a conditional distribution over incident severity given arrival times. To learn the former, we proposed a novel hierarchical clustering approach to extend the use of survival analysis to predict incidents by learning from data the spatial granularity of the model. We used a Multinomial Logistic Regression model to learn the distribution over incident severity. We showed from real traffic accident data from Nashville, TN, USA, that our algorithm results in significant reduction in waiting time for incidents and provides a structured and systematic dispatch policy. We also showed that our prediction model outperforms a state-of-the art incident prediction technique.

CHAPTER VII

Dispatching Emergency Responders

Before introducing the goal of this chapter, it is worth summarizing what we have covered thus far. First, in order to understand how incidents happen in space and time, we constructed generative models that can learn a probability distribution over time to arrival of incidents, conditional on a set of arbitrary features. For any discretized spatial region, this approach can estimate the spatial granularity at which such densities should be learned, and then learn models of incident arrival and severity from historical data. Given such a model, we also looked at optimization approaches to optimally choose locations for two kinds of emergency response - first, we designed a two-stage optimization problem to optimally allocate police responders in anticipation of crimes; and then, we looked at a non-linear optimization formulation to optimally locate stations and the distributions of vehicles (like ambulances and fire trucks) in such stations.

Now, we move to the problem that has the most applicability in practice in an emergency response system - dispatching responders when incidents happen. We point out that this is also the most crucial metric on which any emergency responder system is primarily evaluated. All the algorithms designed so far can be utilized in the field only if responders can actually be guided to incidents faster, thus saving crucial time which can consequently prevent damage to both life and property. The gravity of this problem, therefore, requires that well-defined, well-tested and principled approaches be used for dynamically dispatching responders. Without such an approach, principled algorithmic techniques are often eschewed in practice as delays resulting from ad-hoc dispatch strategies can result in the loss of life (Davis, 2005), and erode the trust in the system.

Hence, we focus on developing a sound decision theoretic framework for continuous-time resource dispatch. We assume that locations and counts of responders and stations are exogenously given, and focus on the dynamic dispatch strategy (The information about responders can either be collected from concerned fire stations, or calculated by using optimization algorithms described in the previous chapters). We formulate the problem as a Semi-Markov Decision Process (SMDP) which evolves in continuous time, and derive an equivalent DTMDP for the formulation. In order to obtain an optimal policy for the SMDP, we propose an algorithm based on policy iteration. We access a simulator to simultaneously simulate our system to estimate values of states, as well as estimate transition probabilities for the DTMDP. We also design efficient heuristic policies leveraging problem structure and domain expertise that can be used to seed the policy iteration algorithm.

We validate our findings by comparing our approach to existing state-of-the-art approaches (Mukhopadhyay et al., 2017) using real traffic data from a major metropolitan city in the US as well as simulated data. Our results demonstrate that our principled approach to dynamic dispatch significantly outperforms the state-of-the-art alternative.

1 Background

1.1 Optimal Static Dispatch

We describe here an approach from prior art that dispatches emergency responders on the basis of an allocation mechanism which has guarantees on wait-times (Mukhopadhyay et al., 2017). We call this policy WTG (Wait-Time Guarantee). The resource allocation algorithm on which WTG is based is a two-fold approach. First, the total area to be serviced is discretized into a set of grids and a probabilistic model of temporal incident arrival is learned for each grid. Then, based on such a model, depots and responders are allocated in space to maximize the total coverage area of responders with bounds on wait times. The algorithm attempts to assign as many grids as possible to specific depots. Based on this allocation, when an incident happens in a grid, WTG first checks if the depot that the grid is assigned to has any free responders or not and dispatches one if available. If not, it looks for other responders returning from service to depots that have no pending incidents. If it fails to locate such responders, the incident enters a waiting queue in the depot that it is assigned to. The algorithm has been shown to outperform existing emergency response systems in a major metropolitan area in USA. We treat this approach as our baseline.

1.2 Semi-Markov Decision Processes

We formally model the problem of dynamic incident response as a semi-Markov decision process (SMDP) (Hu and Yue, 2007). An SMDP is described by the following tuple,

$$\{S, A, p_{ij}(a), t(i,j,a), \rho(i,a), \alpha\}$$

where S is a finite state space, A is the set of actions, $p_{ij}(a)$ is the probability with which the process transitions from state i to state j when action a is taken, $t(i,j,a)$ is a distribution over the time spent during the transition from state i to state j under action a, $\rho(i,a)$ is the reward received when action a is taken in state i, and α is the discount factor for future rewards. In our case, while the state space is finite, it is comprised of a collection of variables, and fully enumerating the state space is not tractable. We consequently leverage a factored state representation described below.

1.3 Dynamic Bayes Networks

Consider a stationary Markov chain representing a dynamical system, with state transition probabilities $P_{ss'}$, and consider a factored representation of states s using a collection of n variables $\{X_1, \ldots, X_n\}$. Let X and X' then be factored state representations for successive states s and s'. A Dynamic Bayes Network (DBN) is a graphical representation of $P_{ss'}$ in factored space, that is, a combination of an acyclic directed graph G_0 on X of intra-state edges, with (i, j) an edge from X_i to X_j, and G_1 a directed graph of inter-state edges on X, X', where direction is from variables (nodes) in X to variables in X', where i, j is an edge from X_i to X'_j. Let the graph G be the union of G_0 and G_1, and for each variable X'_i for a successor state, let $Pa(X'_i)$ be all variables in $X \cup X'$ which have directed edges to X'_i. Then, for each variable X'_i in the successor state we denote it's conditional probability distribution as $P(X'_i|Pa(X'_i))$. The probability distribution for the successor state X' conditional on current state X is then $P_{XX'} = \prod_i P(X'_i|Pa(X'_i))$. In a DBN representation, both the directed acyclic graph G, and the corresponding conditional probabilities $P(X'_i|Pa(X'_i))$ are given.

2 A Continuous-Time Spatio-Temporal Model of Dynamic Emergency Response

Our goal is to develop an approach for making optimal decisions about emergency responder placement and incident response in a dynamic, continuous-time, stochastic environment. We begin with several assumptions on the problem structure and information provided *a priori*. First, we assume that we are given a spatial map broken up into a finite collection of grids G, and assume that we are given an exogenous spatio-temporal model of incident arrival in continuous time over this collection of grids (we describe one such model in Section 2.3). Second, we assume that for each spatial grid cell, the temporal distribution of incidents is homogeneous. This assumption is merely a reflection of the granularity of the spatial discretization: we can in principle always discretize space finely enough so that this assumption approximately holds. Our third assumption is that emergency responders are housed in a fixed and exogenously specified collection of *depots*, each with a pre-defined set of emergency responders. This reflects the relatively long time scale of decisions about the spatial location of the depots themselves.

We assume that when an incident happens, a free responder (if available) is dispatched to the site of the incident. Once dispatched, the time to service consists of two parts: 1) time taken to travel to the scene of the incident, and 2) time taken to attend to the incident. If no free responders are available, then the incident enters a waiting queue; once a responder becomes available, it is then assigned to incidents waiting for service in the order in which they appear in the queue. We can naturally accommodate incidents with different priorities in this framework by using a priority queue. Finally, we assume that the distribution of vehicles in depots is given; it can be calculated using a number of methods in the literature (e.g., (Mukhopadhyay et al., 2017)). We assume that each responder is assigned to a specific depot, and must return to that depot when it is not responding to any incident.

We refer to the entire spatio-temporal process of incident arrival as well as the status of all responders and depots as our *world*. We consider the evolution of this world in continuous time. The dynamics of the world are primarily governed by two events that provide the scope of decision-making : occurrence of traffic incidents and completion of servicing. These are events when responders need to be either sent back to their depots or re-directed to other incidents. Observe that the effect of these actions is not instantaneous. For example, when an incident happens and the action of dispatching a responder is taken, the occurrence of the next state is dependent on the time taken by the concerned responder to travel to the site and attend to the incident. Therefore, given a snapshot of our world (which we refer to as a *state* in our decision making process), the next state depends not only on the given state and the action taken, but also on how *time* evolves between the states.

We model the continuous-time spatio-temporal dynamic decision problem faced by emergency responders using the machinery of Semi-Markov Decision Problems described above. We next describe each of the elements of the SMDP as it captures the emergency responder problem, while also explaining the special structure of this problem which is used for both representing and solving it.

2.1 States

A state at time t is represented by s^t which consists of a tuple $\{I^t, R^t\}$, where I^t is a collection of grid indices that are waiting to be serviced, ordered according to the relative times of incident occurrence. Thus, for any indices j, k and corresponding $i^t_j, i^t_k \in I^t$, $j > k$ implies that incident at grid i^t_j occurred after the incident at grid i^t_k. R^t corresponds to information about the set of responders at time t with $|R^t| = b \forall t$, where b the total number of responders. Each entry $r^t_j \in R^t$ is a set $\{h^t_j, p^t_j, d^t_j, c^t_j\}$, where h^t_j corresponds to the depot that responder j is assigned to, p^t_j is the position of responder j, d^t_j is the destination that it is traveling to (where $d^t_j = 0$ indicates that responder j has no destination assigned), and c^t_j is its current condition, all observed at the state of our world at time t. Observe that $h^t_j, p^t_j, d^t_j \in G \ \forall t, j$. Additionally, $c^t_j \in \{0,1\} \ \forall j, t$, with $c^t_j = 0$ meaning that the responder is currently engaged in service and $c^t_j = 1$ meaning that it is free and available to be dispatched. We acknowledge that while it is not necessary to include h^t_* as a part of the state since responders assignments to depots do not change over time and are exogenously provided, we include this information nonetheless as it makes the states self-containing. Finally, the set of all states is denoted by S.

2.2 Actions

Actions in our world correspond to directing responders either to incidents or back to their depots. We denote the set of all actions by A, with an action by $a^i_{kj} \in A$ corresponding to a decision to send a responder which is from depot k and is currently in grid j, to an incident at grid i. Additionally, we use $A(s^i)$ to denote the set of actions that are available in state $s^i \in S$. In addition, we impose a constraint that whenever responders are available and an incident occurs, we always immediately dispatch *some responder*. We now make several important observations. First, due to the continuous-time nature of our model where a single incident arrives at any point in time, in our model of the world, at most a single action in A is ever taken. This is a result of two model features: 1) since routing a responder to an incident necessarily effects a state transition, and 2) since we always respond to incidents if responders are available, whenever we have multiple available responders, it must be the case that there is at most one new incident to respond to.

2.3 Transitions

We first look at how our *world* evolves before describing transitions. In our model, states evolve between events that provide the scope of decision-making . We refer to these times as decision epochs and such states as decision making states. For convenience, we segregate the two types of events (occurrence of incidents and completion of servicing) and refer to states in which responders become free as completion-states S_c and states in which incidents occur as incident-states S_a. We also observe that states can evolve between decision epochs, and every such change in state does not present a chance to make decisions (i.e., the corresponding $A(s)$ is empty). As an example, responders going back to their depot move through different grids, which updates the state variable R, but presents no scope for decision making in our process, unless an incident happens. We also make the assumption that no two events (incident or completion of servicing) can occur simultaneously in our world. In case such a scenario arises, since the world evolves in continuous time, we can add an infinitesimally small time interval to segregate the two events and create two separate states.

In order to segregate decision making states from other states, we divide the model of our world into two processes as in prior work (Puterman, 2014): an embedded MDP that is observed only at decision epochs, and a natural process that evolves as if it is observed continuously through time, but presents no relevant information to the decision maker, unless the world is at a decision epoch. Since the decision making process is essentially the natural process observed at special (decision making) instances, the two processes are always the same at decision epochs. This segregation helps us in two ways: first, it truncates the state space by only looking at states that are relevant from the perspective of decision-making, and more importantly, it lets us remain agnostic about how the natural process evolves, thereby letting us exploit well-established models for transitions between our decision epochs.

Having described the evolution of our *world*, we now look at both the transition time between events, as well as the probability of observing a state, given the last state and action taken. We define the former first, denoting the time between two states s^i and s^j by the random variable t_{ij}.

Since decision epochs are governed by the occurrence of incidents and service completions, we first describe our models between these events. We denote the time between incidents by the random variable t^a and time to service an incident by t^s. We model inter-arrival times between incidents by using survival-analysis, that has been widely used to model time to events, and has recently been used to forecast urban incidents (Mukhopadhyay et al., 2016a, 2017), making it a natural candidate for our purposes. Survival analysis is a broad class of methods that are used to model the distribution of time and risk for events. We use the accelerated time effect (AFT) model in which covariates increase or decrease the expected time to next incident (Miller Jr, 2011), letting us directly model time, rather than risk. Formally, we model the time t between successive incidents as $f(t|w)$, where t follows an exponential distribution, and w is a set of arbitrary features that affect t. As in standard AFT models, we model the arrival rate for the exponential distribution as a log-linear model in terms of the features w. Thus, the probability distribution of time to next incident t^a in a given spatial grid is

$$f(t^a|w) = \lambda^a e^{-\lambda t^a} \tag{VII.1}$$

where $\log \lambda^a = \sum_{w_j \in W} \theta_j w_j$ and θ are the regression coefficients learned from data. This form of the model is particularly useful to us in simulating our world, the purpose of which we explain in Section 3. As in prior work (Mukhopadhyay et al., 2017), we model service time t^s by an exponential distribution

$$f(t^s) = \lambda^s e^{-\lambda t^s}.$$

In combination, our models of t^a and t^s are crucial as they induce memoryless arrival and service times which allow us to model the entire dynamic dispatch process as an SMDP with special structure.

To understand the temporal transitions, we first explain how time evolves between two states by considering the following scenarios. First, consider a series of two events as shown in Fig. VII.1, where at time t_1, an incident happens at grid i (state s^{t1}) and a vehicle is dispatched from grid j, and at time t_2, the vehicle finishes servicing the incident (state s^{t_2}). The time between these two states is $d_{t_1 t_2} v + t^s$, where $d_{t_1 t_2}$ is the distance between the concerned grids at states s^{t_1} and s^{t_2} and v is the (known) velocity of the responders. Now, let us consider the scenario that an incident or completion of service happens at time t_3, resulting in a new state s^{t_3}, such that $t_1 < t_3 < t_2$. The time between states s^{t_3} and s^{t_2} now depends on whether the responder traveling to grid j had reached its destination or not. If it had not, we can calculate the distance left for it to travel and hence the time $t_{t_3 t_2}$ based on information available in s^{t_3}. In case it had reached and started servicing, we can leverage memorylessness of the service time distribution and *reset* the remaining service time, thereby estimating $t_{t_3 t_2}$. Next, suppose that an incident happens at time t_1 (state s^{t_1}), and the next incident occurs at time t_2 (state s^{t_2}). The transition between incidents can be modeled directly by the survival model in Eq. VII.1. Now, imagine that a new event (service completion, state s^{t_3}) happens at time t_3 such that $t_1 < t_3 < t_2$, as shown in Fig. VII.2. Observe that since incident arrivals are also memoryless, the time to state s^{t_2}, as seen from state s^{t_3} can again be reset.

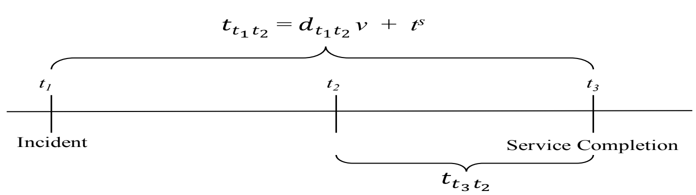


Figure VII.1: Transition Times when $s^j \in S_c$

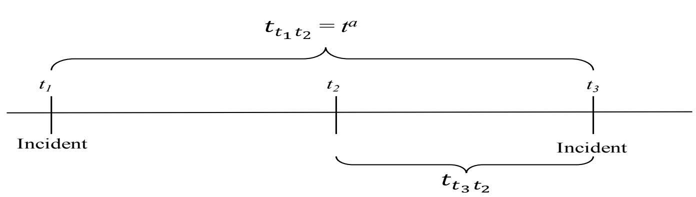


Figure VII.2: Transition Times when $s^j \in S_a$

To summarize, for any two states s^i and s^j

$$t_{ij} = \begin{cases} \sum_{k=1}^{b} \mathbb{I}\{(c_k^j - c_k^i) > 0\} d_{ij} v + t_s \text{ if } s^j \in S_c \\ \\ t_{ij}^a \text{ if } s^j \in S_a. \end{cases} \tag{VII.2}$$

Having considered the temporal evolution between states, we now consider the probability of observing a state at a decision epoch, given a particular state at the last decision epoch and the associated action. We denote the probability that the process moves from state s^i (at a decision epoch) to state s^j (in the next decision epoch) under action a by $p_{ij}(a)$, where states belong to the embedded MDP. Note that the probability of transition between these states cannot solely be modeled using the incident and service time distributions $f(t^a)$ and $f(t^s)$, as the natural process evolves between decision epochs. $p_{ij}(a)$ thus captures the transition probabilities for states between decision epochs while taking into account the implicit evolution of the natural process.

For a compact representation of the state transition distribution p, we use a Dynamic Bayes Network (DBN) to leverage conditional independence relationships among the state variables. The structure of the DBN is visualized in Fig. VII.3. Specifically,

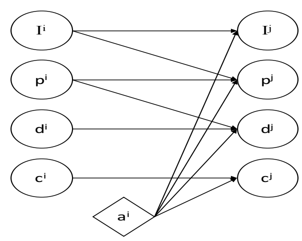


Figure VII.3: Dynamic Bayes Network to model Inter-State Transition

$$p_{ij}(a) = P(I^j|I^i,a^i)P(p^j|p^i,I^i,a^i)P(d^j|d^i,p^i,a^i)P(c^j|c^i,a^i) \tag{VII.3}$$

where a^i is the action taken at state s^i. While the structure of the DBN is self-explanatory, we point out some of the key insights that leverage the structure of the responder dispatch problem. For predicting incidents, we use the standalone survival model which captures all relevant information for predicting future incidents. Consequently, future incidents are conditionally independent of the other state variables, given current incidents. However, pending incidents are removed from states as they are serviced; hence, the transition function for I is dependent on the action taken.

2.4 Rewards

Rewards in SMDP usually have two components: a lump sum instantaneous reward for taking actions, and a continuous time reward as the process evolves. Our system only involves the former, which we denote by $\rho(s,a)$, for taking action a in state s. Observe that the best (myopic) scenario in emergency response is when a responder is already present at the scene of the incident, while the worst scenario occurs when the only available responder has to travel a distance equal to the largest possible distance in the given area to attend to an incident. We design our reward structure in accordance with these observations. When a responder is dispatched in state s^i to attend to an incident that results in a state completion state s^j, we denote the reward as $d_{max} - d_{ij}$, where d_{max} is the largest possible distance in the given area under consideration and d_{ij} is the distance between the concerned grids. Also, we assume that the action of sending a responder back to its depot has 0 reward, since this is the default action that must be taken if an incident is not waiting to be serviced.

2.5 Dispatch Process and Decision Problem

In summary, the evolution of the responder dispatch world happens as follows:

1. Once the system is in state s^i, an action $a \in A(s^i)$ is taken.
2. The system receives an instantaneous reward $\rho(s^i, a)$.
3. The system transitions to state s^j according to the probability distribution $p_{ij}(a)$
4. The system takes time t to make the transition, where $t \sim t_{ij}$

Having described the structure of the SMDP representing dynamic continuous-time spatio-temporal problem of emergency response, we proceed to outline the general goal in solving it. Any solution to this problem is to obtain a policy π, which for any given state s^i prescribes an action $\pi(s^i)$ to be taken in that state. Ideally, the policy should produce an *optimal*, i.e., utility-maximizing, action a, with the notion and form of utility defined beforehand. Formally, for any arbitrary state s^i, we define the expected discounted total reward over an infinite horizon as

$$V^{\pi}(i) = \sum_{n=0}^{\infty} \mathbb{E}\{e^{-\alpha T_n} \rho(s^n, \pi(s^n))\}$$

where s^n is the state at n^{th} decision epoch, and T_n its duration. Our goal is to find the optimal policy π^* which, starting from for an arbitrary state i, maximizes the sum of expected discounted rewards, with a minor caveat. Emergency responders, in practice, are often governed by instructions to send the responders that are close to the scene of an incident. Although this is myopic, it provides the best chances of dealing with the current incident at hand, and failing to do so might result in immediate damage and/or loss of life. We take this into effect by adding a constraint to our optimization problem as follows

$$\begin{aligned} &\sup\nolimits_{\pi} V^{\pi}(i) \\ &s.t \\ &\rho(s, \pi(s)) \geq \gamma \rho(s, a) \, \forall a \in A(s) \, \forall s \in S. \end{aligned} \tag{VII.4}$$

The constraint enforces that the immediate reward taken at a step is at least γ times the best reward, where γ is a user-defined parameter based on the nature of the specific emergency response system.

3 Solution Approach

We now present an approach for computing an optimal policy of the formulated SMDP model of dynamic emergency response. Our approach is based on policy iteration (Puterman, 2014), which is a two-step process for computing an optimal policy for an MDP by iteratively improving upon a starting seed policy until convergence. In the first step, the values of states are estimated under a fixed policy from the last iteration. The second *policy improvement* step then incrementally improves upon the previous policy by finding a better action in each step.

Unlike conventional MDPs with a small state space, we face two challenges that make direct policy iteration impractical. First, we have a combinatorially large state space. Second, the state transition probabilities $p_{ij}(a)$ are unknown; rather, we can simulate transitions and use such simulations to estimate transition probabilities. Our approach below addresses these problems.

3.1 Discretization

We first present an approach that assumes that the state transition probabilities are known. We subsequently relax this assumption.

A general approach of solving an SMDP is to derive an equivalent discrete-time MDP (DTMDP) (Hu and Yue, 2007), and then solve the DTMDP by standard techniques like policy iteration. Before presenting the conversion, we introduce some additional notation. We denote by F the cumulative distribution function for the random variable t, that is used to model the transition time between states. To begin with, for each pair of states s^i and s^j, and action a, we define

$$\beta_\alpha(i,j,a) = \int_0^\infty e^{-\alpha t} F_{ij}^a(dt). \tag{VII.5}$$

Using Eq. VII.5, we then define the *expected discount factor* as

$$\beta_\alpha(i,a) = \sum_j p_{ij}(a)\beta_\alpha(i,j,a)$$

Intuitively, $\beta_\alpha(i,a)$ measures the significance of one unit of reward obtained at the current decision epoch, valued at the next decision epoch, when the continuous time discount factor is α. Finally, we define $\beta_\alpha = \sup_{i,a}\beta_\alpha(i,a)$.

As mentioned earlier, transition times between two states s^i and s^j depend on whether $s^j \in S_a$ or $s^j \in S_c$. We look at these two cases separately.

Case I: When $j \in S_c$, from equation VII.2

$$t = c(i,j) + t^s \Rightarrow t^s = t - c(i,j) \tag{VII.6}$$

where $c(i,j) = \sum_{k=1}^{b} \mathbb{I}\{(c_k^j - c_k^i) > 0\} d_{ij} v$, which given two states, is a constant. Let $g(t)$ represent the density function of the random variable t, and $f(t^s)$ represent the density of the service time distribution, as mentioned earlier. Now,

$$g(t) = f(t^s) = f(t - c(i,j)) = \lambda_s e^{-\lambda_s (t - c(i,j))} \tag{VII.7}$$

and

$$F(t) = \int_{c(i,j)}^{t} \lambda_s e^{-\lambda_s (t - c(i,j))} dt.$$

Then,

$$\begin{aligned} \beta_\alpha(i,a,j) &= \int_{c(i,j)}^{\infty} e^{-\alpha t} \lambda_s e^{-\lambda_s (t - c(i,j))} dt \\ &= \frac{\lambda_s}{\lambda_s + \alpha} e^{-\alpha c(i,j)}. \end{aligned} \tag{VII.8}$$

Case II: When $j \in S_a$, let the time from the last incident to s_i be τ. Since t^a is distributed exponentially,

$$g(t > x) = f(t^a > \tau + x | t^a > \tau)$$

Hence, similar to Case I,

$$\beta_\alpha(i,a,j) = \frac{\lambda_{ij}^a}{\alpha + \lambda_{ij}^a}.$$

Having described the necessary transformations, we define the corresponding Discrete-time Markov Decision Process (DTMDP) as $\{S, A, \bar{p}_{ij}, \rho, V_\beta, \beta_\alpha\}$, where $\bar{p}_{ij}(a) = \beta_\alpha^{-1} \beta_\alpha(i,a,j) p_{ij}(a)$ is the scaled probability state transition function and β_α is the updated discount factor. The value of a state s^i in the transformed MDP can then be represented as

$$V_\beta(s^i) = \sup_a \{\rho(i,a) + \sum_j \beta_\alpha(i,a,j) p_{ij}(a) V_\alpha(j)$$

This DTMDP is equivalent to the original SMDP according to the total reward criterion (for the proof of this equivalence, see Hu and Yue (Hu and Yue, 2007)).

3.2 SimTrans : Simulate and Transform

The transformed DTMDP still suffers from the two technical difficulties discussed above. We now proceed to address these through a novel algorithm. We call this algorithm *SimTrans*, as it combines **sim**ulating a generative model and **trans**forming it into an equivalent DTMDP, in order to solve a SMDP formulation.

The basic idea is to approach policy iteration by simulating the world to estimate values of states with one important addition. Choosing an optimal action in a state when given access to a simulator has been previously explored by Kearns et al. (Kearns et al., 2002), Péret and Garcia (Péret and Garcia, 2004). SimTrans accesses the simulator to estimate a state's value but at the same time, iteratively builds confident estimates of the state transition probabilities, which can then be used for the transformed DTMDP. Once such estimates are available for any state-action pair, the algorithm chooses to accept such an estimate and avoids simulating, thus reducing computational costs. We present SimTrans in Algorithm 8.

The algorithm presents standard policy iteration with one modification, a procedure $ESTVAL$, that is added to estimate values of states. We start with a subset of states and, given an arbitrary state s^i and a policy π, SimTrans decides whether to simulate the world to estimate $V^{\pi}(s^i)$ or access pre-computed transition probabilities and directly calculate $V_{\beta}(s^i)$. To do this, *ESTVAL* first accesses a procedure $conf(s^i, \pi(s^i)$ (refer operation 8 in Algorithm 8) to check if it has access to *confident* estimates of $p_{ij}(\pi(s^i)\ \forall j \in R(i)$, where $R(i)$ represents the set of possible next states from s^i (we formally define the notion of *confidence* later). If such estimates are available (refer to operation 9), the algorithm moves on to find the expected discounted rewards from future states, the expectation being taken with respect to $\hat{p}_{ij}(\pi(s^i))$ (the estimated value of $p_{ij}(\pi(s^i))$). In case such estimates are not available, the algorithm estimates $V^{\pi}(s^i)$ by a direct Monte-Carlo estimation approach. It simulates the world m times under policy π, starting from s^i, where m is the user-defined Monte-Carlo budget (refer operation 12). At any iteration l of the simulation, we obtain an estimate $\hat{V}_l^{\pi}(s^i)$. The final estimate is simply calculated as the sample mean of the estimates. Thus, $V^{\pi}(s^i) = \frac{\sum_{l=i}^{m} \hat{V}_l^{\pi}(s^i)}{m}$. Also, every time the world is simulated, the algorithm tracks the state transitions generated (referred by ϕ in SimTrans) and updates estimates of state transition probabilities (refer operation 14). Throughout the process, SimTrans only looks at actions that are at least γ times the best myopic action available.

We now look at how the procedure checks confidence in estimates generated by the algorithm. We point out that while any notion of statistical *confidence* can be used in the algorithm, we choose the standard two-sided confidence bound, with two user-defined parameters. The algorithm takes as input a tolerance bandwidth ω and a confidence parameter r. For any arbitrary $p(.)$, given a series of its estimates, the algorithm calculates with confidence r, the interval $\tilde{p}$ within which the true parameter lies as $\tilde{p} = \frac{2Z_r s}{\sqrt{n-1}}$, where s is the sample deviation, n is the number of available samples and Z_r is the critical value of the standard normal for confidence r. If $\tilde{p} \leq \omega$, the algorithm accepts the estimates; otherwise, it simulates the world to get an estimation.

Algorithm 7 SimTrans

INPUT: Initial Policy π_0, Initial states S_0, Maximum Iterations MAX_ITER, Confidence Parameters ω, r
OUTPUT: Final Policy π^*
for $l = 1..MAX\_ITER$ **do**
  Policy Evaluation:
  for $i \in S_{l-1}$ **do**
    $V_l^{\pi_{l-1}}(i) = \text{EstVal}(\pi, i, b, m, l)$
  end for
  Policy Improvement:
  for $i \in S_{l-1}$ **do**
    if $\text{conf}(i, \pi(i)) \leq \omega$ **then**
      $\pi_l(i) = \text{argmax}_{a \in A(i)} \rho(i,a) +$
        $\sum_{j \in R(i))} \hat{p}_{ij}(\pi_{l-1}(i)) \beta_\alpha(i,a,j)\, \text{EstVal}(\pi_{l-1}, j, b, m, l)$
    else
      for $a \in A(i)$ **do**
        $\pi' = \pi_{l-1}$
        $\pi'(i) = a$
        $V^a(i) = \text{EstVal}(\pi', i, b, m, l)$
      end for
      $\pi_l(i) =_a V^a(i)$
    end if
  end for
  S_l = UpdateStates(S_{l-1})
end for

Having looked at the approach that combines simulation and discretization, we now address the problem of an extremely large state space. To address this issue, we take two measures. First, we start with an initial subset of states and gradually add states as we simulate the world. Second, we seed our policy iteration algorithm with heuristic policies, that are designed based on prior work (Mukhopadhyay et al., 2017) and domain expertise. Thus, for any state, we always have access to a default policy.

We consider two seed policies. The first is WTG, which is based on a static assignment of depots to spatial service grids, described in Section 1. The second is a novel heuristic policy, Multiple Depot Heuristic (MDH) that addresses the two shortcomings of WTG. First, although the placement algorithm of WTG is built on guarantees on upper bounds on wait times, the dispatch algorithm,

Algorithm 8 Procedure *EstVal*

```
procedure EstVal(π, i, b, m, l)
    if Available(V_l^π(i)) then
        return V_l^π(i)
    end if
    if b=0 then
        return ρ(i, π(i))
    end if
    if conf(i, π(i)) ≤ ω then
        V^π(i) = r_α(i, π(i)) +
            ∑_{j∈R(i))} p_ij(π(i))β_α(i, a, j) EstVal(π, j, b − 1, m, l)
    else
        for k = 1..m do
            V̂_k^π(i), φ_k = Simulate(π, i, b)
        end for
        UpdateP(φ_1, ..., φ_m)
        V^π(i) = (∑_{k=1}^{k=m} V̂_k^π(i)) / m
    end if
    return V^π(i)
end procedure
```

Algorithm 9 Multi Depot Heuristic

```
INPUT: Grid i, State s^t, Allocation Alloc
OUTPUT: π(s^t) : Policy for the current state
Let j = Alloc(i)
count = Free(j)
if count > 0 then
    return a^i_jj
end if
RespSort = Sort(R^t, i)
for r ∈ RespSort do:
    if Pending(h_r^t) = 0 then
        return a^i_{h_r^t p_r^t}
    end if
end for
```

in practice, does not guarantee bounds on wait times for all incidents. This happens since some grids (say grid i) are not assigned to any depots and the nearest depot (say depot j) is contacted in case incidents occur in such grids. This causes wait time bounds to marginally fail for depot j (as responding to grid i was not a part of the resource placement algorithm and is enforced on the depot during dispatch). Second, domain expertise dictates that while cross-depot dispatch should be reduced (for vehicle wear and tear as well as other maintenance issues), it should not be completely ignored. We present this dispatch policy formally in Algorithm 9. When an incident happens in grid i in state s^t, MDH first looks at the depot that grid i is assigned to (refer operation 3) according to a responder placement algorithm, such as Mukhopadhyay et al. (Mukhopadhyay et al., 2017), Silva and Serra (Silva and Serra, 2008). Mukhopadhyay et al. (Mukhopadhyay et al., 2017) also looks at vehicles returning after serving incidents). In case no free responders are available, it sorts all free responders available on the basis of proximity to the scene of the incident (refer operation 8). It then iteratively checks each responder, and sends one from a depot that has no pending calls. While this approach honors the fixed assignment of depots to grids, it accesses cross-depot dispatch in some cases, when the assigned depot is unable to service.

4 Experimental Evaluation

4.1 Data

Our evaluation uses traffic accident data and assault data obtained from the *fire* and *police* departments in a medium-size city in the US, with a population of approximately 700,000. For this fire department, traffic accidents and crimes requiring ambulance services comprise a large majority of incidents it responds to (fires, in contrast, are relatively rare). We looked at traffic accident data for 26 months, from 2014 - 2016, comprising of a total of 19,373 traffic accidents, and assault data for the year 2014, consisting of a total of 7,100 incidents. Each accident is accompanied by its time of occurrence, the time at which the first responding vehicle reached the scene and the time at which the last responding vehicle was back at service, which refers to completion of servicing an incident. We use the same incident prediction model as in prior work (Mukhopadhyay et al., 2017). We also produce two synthetic datasets using our incident prediction model, by scaling the incident arrival rate in the exponential distribution by 0.5 and 2. This provides us with a test bed to evaluate the model on potential urban areas that are different than the one in our dataset.

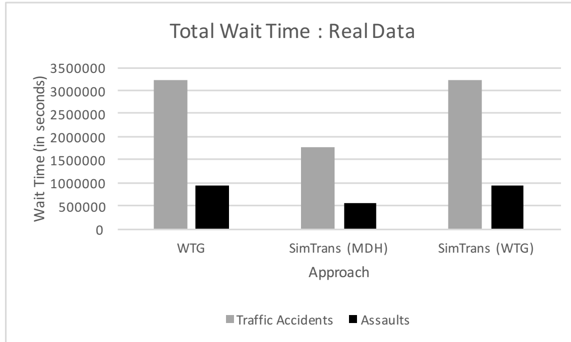


Figure VII.4: Response Times : Real Data (Lower is better)

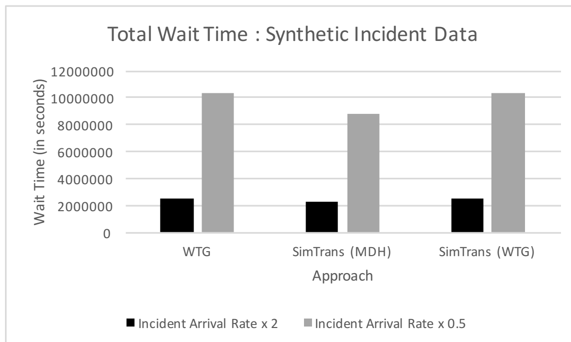


Figure VII.5: Response Times : Synthetic Data (Lower is better)

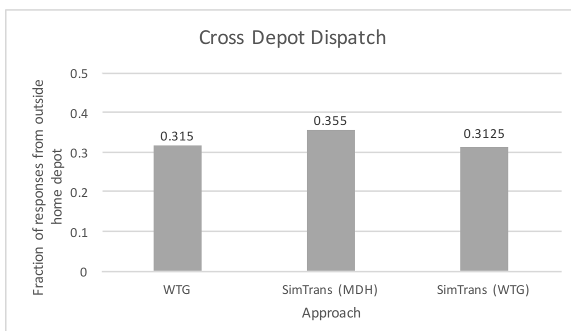


Figure VII.6: Cross Depot Dispatch

4.2 Results

We evaluate the proposed solution by a direct comparison of wait times for incidents. We set $\omega = 0.1$ and $r = 0.95$ for our experiments. As a constraint on the action space, we set $\gamma = 0.95$, since quick emergency response by ambulances is critical to saving lives. This means that, for any state, SimTrans only looks at actions that provide at least 0.95 times the immediate reward of the best available action. For the complete set of incidents, we dispatch responders based on standalone WTG and SimTrans. We test the performance of SimTrans by first using WTG (SimWTG) as a seeding policy, and then MDH (SimMDH). We present the results in Fig. VII.4. The results are shown after 3 iterations of SimTrans. We see that SimMDH reduces wait times by almost 50% for both traffic accidents and assaults. SimWTG, on the other hand, outperforms standalone WTG only marginally on both types of incidents.

The results on synthetic dataset are shown in Fig. VII.5. In this case, SimWTG edges out WTG marginally, while SimMDH again shows a remarkable reduction in wait times. The overall wait times for synthetic data are slightly higher than real data as the generative model assigns non-zero probability to grids that are unlikely to see traffic accidents. These grids are rarely sampled, but the presence of incidents in such grids drives wait times higher. In order to analyze the reduction in wait times, we check for each algorithm the percentage of incidents that are served from assigned depots (Fig. VII.6). We see that SimMDH increases cross depot dispatch by over 4%, thereby increasing the availability of responders to travel shorter distances when they need to.

5 Conclusion

We proposed a principled decision-theoretic framework to address the problem of emergency responder dispatch in a continuous-time, dynamic and stochastic environment. We framed the problem as a Semi-Markov Decision Process leveraging insights from the problem structure. We used a well established incident prediction model and a Dynamic Bayes Net to create a factored and compact representation of the state transitions. Then, we proposed a novel algorithm to solve it, that simulates the environment and simultaneously learns state-transition probabilities. Also, we designed efficient heuristics to seed our proposed algorithm. We evaluated our algorithm on both real data from Nashville as well as synthetic data and showed that our model outperforms previous state-of-the-art.

CHAPTER VIII

Online Approaches to Aid Emergency Response

1 Introduction

The problem of dispatching emergency responders is intrinsically hard. Emergency responders work under legal and moral constraints - for obvious reasons, high immediate utility from a quick response to a current incident takes preference over delayed rewards. Also, urban areas are dynamic and complex environments. This has two important consequences. First, any model of an urban environment that captures sufficient information from an urban area needs to be elaborate and detailed. Second, due to the dynamic nature of the environment, static approaches learned over a specific model of environment quickly become stale. This motivates us to look at approaches not only capture enough environmental dynamics, but are quick to train and adaptable. In chapter VII, we formulated the emergency dispatch problem as a Semi-Markov Decision Process (SMDP), and created an algorithmic approach that can solve it optimally. However, such an approach takes weeks to train on a specific model of the environment. Naturally, the environmental dynamics change by the time the training completes. This severely limits the scope of deployment of such an approach in practice. In this chapter, we focus on two online models to alleviate this concern [1]. First, we create a model of incident prediction that can absorb streaming data, which lets us adapt the incident prediction model dynamically. Second, we shift our focus from finding a policy for the above-mentioned SMDP and seek to find near-optimal action for a given state of the system. We first explain the models, and then show how they perform significantly better than prior work, both in terms of accuracy and computational running time.

2 Real Time Incident Prediction

Our goal with incident prediction is still the same - we want to learn a probability distribution over incident arrival in space and time, and we continue with our model of parametric survival models (Mukhopadhyay et al., 2016b, 2017, 2018). Our problem setup is the same as well. We consider that the entire urban area is divided into as set of grids G. Incidents happen in these grids with an inter-arrival temporal distribution f.

Formally, the survival model is $f(t|\gamma(w))$, where f is a probability distribution for a continuous random variable τ representing the inter-arrival time, which typically depends on covariates

[1] Geoffrey Pettet had equal contribution in this work

w via the function γ. The model parameters can be estimated by the principled procedure of Maximum Likelihood Estimation (MLE). The spatial granularity at which such models are learned can be specified exogenously - a system designer can choose to learn a separate f for each discretized spatial entity (grids in our case), learn one single model for all the grids or learn the spatial granularity from data itself. This choice is orthogonal to the approach described in this paper and we refer interested readers to our prior work (Mukhopadhyay et al., 2017) for a discussion about such models.

We shift our focus directly to survival models that are used to learn $f(\tau|\gamma(w))$. Intuitively, given an incident and a set of features, we want to predict when the next incident might happen. Before proceeding, we reiterate that we an added piece of notation - we assume the availability of a dataset $D = \{(x_1, w_1), (x_2, w_2), .., (x_n, w_n)\}$, where x_i represents the time of occurrence of the i^{th} incident and w_i represents a vector of arbitrary features associated with the said incident. The random variable τ is used to measure the inter-arrival time between incidents, and we use τ_i to represent the time interval $x_{i+1} - x_i$. The function γ is usually logarithmic and the relationship of the random variable τ with the covariates can be captured by a log-linear model. Formally, for a time-interval τ_i and associated feature vector w_i, this relationship is represented as

$$log(\tau_i) = \beta_1 w_{i1} + \beta_2 w_{i2} + ... + \beta_m w_{im} + y \tag{VIII.1}$$

where, $\beta \in \mathbb{R}^m$ represents the regression coefficients and y is the error term, distributed according to the distribution h. The specific distribution of f is actually decided by how the error y is modeled. We choose to model τ by an exponential distribution Thus, in our incident prediction model, we assume that h takes the following form

$$h_Y(y) = e^{y - e^y}$$

Using equation VIII.1, for a given set of incidents, the log-likelihood of the observed data under the specific model can be expressed as

$$L = \sum_{i=1}^{n} \log h(\tau_i - w_i^T \beta) \tag{VIII.2}$$

The standard way to estimate the parameters of the model is to use a gradient-based iterative approach like the Newton-Raphson algorithm, yielding a set of coefficients β^* that maximize the likelihood expression. Since such a model is generative, it can be used to simulate chains of incidents, which is particularly helpful in building a simulator, the purpose of which we explain in the next section.

As pointed out before, such an approach is *offline*. However, it is imperative to capture the latest trends in incident arrival to accurately predict future incidents, which motivates us to design an online approach for learning and predicting incidents. We introduce some added notation before describing the algorithmic approach. First, we reiterate that β^* is used to refer to coefficients already learned from dataset D. Further, we assume that a new set of incidents $D^{'} = \{(x_1^{'}, w_1), (x_2^{'}, w_2), .., (x_k^{'}, w_k)\}$ is available that consists of incidents that have happened after (in time) the original set of incidents. We aim to update the regression coefficients β using $D^{'}$, assuming that the model already has access to β^*.

In order to address this problem, we use stochastic gradient descent to update the distribution f in an online fashion. Formally, we start with the known coefficients β^* and, at any iteration p of the process, we use the following update rule

$$\beta^{p+1} = \beta^p + \alpha \nabla L(\beta^p, D^{'})$$

where $\nabla(L(\beta^*, D^{'})$ is the gradient of the log-likelihood function calculated using $D^{'}$ at β^p and α is the standard step-size parameter for gradient based algorithms. Using equation VIII.2, likelihood of the incidents in the dataset $D^{'}$ can be represented by

$$L = \sum_{i=1}^{k} \log e^{(\log t_i - \beta^* w) - e^{(\log t_i - \beta^* w)}}$$

and subsequently,

$$\frac{\partial L}{\partial \beta_j} = \sum_{i=1}^{k} -w_{ij} + w_{ij}\{e^{(\log t_i - \beta^* w_i)}\}$$

The update step is repeated until improvements in the likelihood of the available data. Having already summarized the important steps in the algorithm in this section, we present it formally in Algorithm 10.

This mechanism enables us to update the incident prediction model in an online manner, saving vital computation time for the responder dispatch system. Also, this implicitly betters the dispatch algorithm by generating incident chains that capture the latest trend in incident occurrence.

Algorithm 10 Streaming Survival Analysis

```
1: INPUT: Regression Coefficients β*, Dataset D', Tolerance α, Likelihood Function L, Maximum
   Iterations MAX_ITER
2: for p = 1..MAX_ITER do
3:     β^{p+1} = β^p + α∇L(β^p, D')
4:     if L(β^{p+1}, D') < L(β^p, D') then
5:         Return β^p
6:     end if
7: end for
8: Return β^p
```

3 Dispatch Algorithm

We begin the discussion on our dispatch algorithm by first reiterating how the dispatch problem can be modeled as an SMDP, and solved by canonical policy iteration. We present a high-level overview of this approach here, and refer interested readers to chapter VII for details. The first step in the process is to convert the SMDP to a discrete time MDP M_d, which can be represented as

$$\{S, A, \bar{p}_{ij}, \rho, V_\beta, \beta_\alpha\}$$

where $\bar{p}_{ij}(a) = \beta_\alpha^{-1}\beta_\alpha(i,a,j)p_{ij}(a)$ is the scaled probability state transition function and β_α is the updated discount factor. We point out that the transformed MDP is equivalent to the original MDP according to the total rewards criterion (Hu and Yue, 2007), and hence it suffices to learn a policy for M_d. Given such a conversion, the approach to solving the MDP involves accessing a simulator to learn the state transition probabilities for M_d (Mukhopadhyay et al., 2018). The algorithm, *Sim-Trans*, an acronym for *Simulate and Transform*, uses canonical Policy Iteration on the transformed MDP M_d, with an added computation. In order to estimate values (utilities) of states, the algorithm simulates the entire system of incident occurrence and responder dispatch and keeps track of all states transitions and actions, and gradually builds statistically confident estimates of the transition probabilities. Once confident estimates are available (with the measure of statistical confidence defined appropriately), the algorithm accesses such estimates and avoids simulations, thereby saving computation time. Since policy iterations are guaranteed to converge to the optimal policy, *Sim-Trans* essentially finds a close approximation of the optimal policy, assuming that the estimates of the transition probabilities learned are close to the true probabilities.

This process, however, is extremely slow and fails to work in dynamic environments. In order to tackle this problem, we first highlight an important observation - one need not find an optimal action for each state as part of the solution approach. At any point in time, only one decision-

Algorithm 11 Real-Time SMDP Approximation Main Procedure

1: **INPUT**: State s, Current Environment E, Horizon h, Stochastic Horizon h^s, Simulation Budget b, Generative Model Θ
2: Set current depth $d \leftarrow 0$
3: $C \leftarrow b$ incident chains generated by $\Theta(E)$
4: Set Scores $U \leftarrow \varnothing$
5: $\bar{A}$ = *SelectCandidateActions* (s, d, h^s)
6: **for** incident chain $c \in C$ **do**
7:     $u \leftarrow$ *ChainEvaluation* $(c, s, d, \bar{A}, h^s, h)$
8:     **for** candidate action $a \in \bar{A}$ **do**
9:         $U[a] \leftarrow U[a] + u[a]$
10:     **end for**
11: **end for**
12: Return $\text{argmin}_{a \in \bar{A}}(U[a])$

making state might arise that requires an optimal action. This difference is crucial, as it lets us bypass the need to learn an optimal policy for the entire MDP. Instead, we describe a principled approach that evaluates different actions at a given state, and selects one that is sufficiently close to the optimal action. We do this using sparse sampling, which creates a sub-MDP around the neighborhood of the given state and then searches that neighborhood for an action. In order to actualize this, we use Monte-Carlo Tree Search (MCTS).

Another important observation is that the incident prediction model discussed in section 2 is generative and independent of dispatch decisions, which lets us simulate incidents independently. Models of travel (Mukhopadhyay et al., 2019) as well as service times for responders can also be learned from data, the entire urban area can therefore be simulated. We denote such a simulator by Θ, which can *generate* samples of how the urban area evolves, even though the exact state-transition probabilities are unknown. This observation lets us simulate future states from a given state, leading to the creation of a state-action tree as shown in Fig. VIII.1. We refer to the algorithm as Real-Time SMDP Approximation and explain how it is designed next. Through the course of this discussion, we assume that the simulator can access a modular (and possibly exogenously specified) model to predict the environment at any point in time.

Algorithms 11, 12, 13, and 14 describe the various functions of our MCTS approach. We start our discussion with Algorithm 11, which is the highest level procedure that is invoked when presented with a decision-making state. First, b incident chains are sampled using the generative model Θ (refer to step 3 in Algorithm 11), where each chain is a time ordered list of sampled incidents. We create multiple chains in order to limit the impact of variance in the generative model. Next, the algorithm starts building the MCTS tree. We use the function $node(s, \eta, d, t)$ to refer to the creation of

Algorithm 12 Select Candidate Actions for Given State

procedure SELECTCANDIDATEACTIONS(State s, Depth d, Stochastic Horizon h^s)
    $A^s \leftarrow$ set of available actions in state s
    $a^* = \text{argmin}_{a \in A^s}(\rho(s,a))$
    if depth $d \geq h^s$ **then**
        Return a^*
    else$\bar{A}^s = \{a | a \in A^s \text{ and } \rho(s,a) \leq \epsilon * \rho(s,a^*)\}$
        Return $\bar{A}^s$
    end if
end procedure

Algorithm 13 Evaluate a Chain of Incidents

procedure CHAINEVALUATION(Incident Chain c, State s, Depth d, Candidate Actions $\bar{A}$, Stochastic Horizon h^s, Horizon h)
    Set scores $u \leftarrow \varnothing \quad d \leftarrow d+1$
    for action $a \in \bar{A}$ **do**
        Next state $s' \leftarrow UpdateState(s,a,c)$
        Utility util $= s'$.responseTime
        Root $\leftarrow$ new Node(State = s', util = util)
        Update $u[a] \leftarrow CreateStateTree(Root, c, d, h^s, h)$
    end for
    Return u
end procedure

a node in the tree, which tracks the current state of the system (s), the cost of the path from the root to the node (η), the depth of the tree (d) and the total time elapsed (t). Also, we use *UpdateState(s,a,c)* to retrieve the next state of the system, given the current state s, action a and chain c. For any state, we start by finding a set of candidate actions for the given incident (refer to step 5 in Algorithm 11), which takes the algorithmic flow to Algorithm 12. The candidate actions are chosen according to the current depth of the MCTS tree - if the tree is within the stochastic horizon h^s, the candidate actions include all actions with a cost that is at most ϵ times the cost of the myopically optimal action a^*. The parameter ϵ can be varied to control the trade off between the computational load of the algorithm and performance. Once the tree is deeper than h^s, the algorithm picks the best myopic action as a heuristic to construct the tree's nodes until depth h, since rewards are sufficiently discounted. After candidate actions are found for the sampled incidents of the chain, Algorithm 13 is used to evaluate possible decision-making courses - each available action is tried and the MDP is simulated to generate future decision-making states, from which the entire process is repeated. This gradually builds a tree, where each edge is an action and each node is a decision-making state. We explain this procedure in Algorithm 14.

The key steps of the procedure are as follows. First, costs are tracked for every branch as the tree is built (refer to steps 10 in Algorithm 14), which is based on the response time in seconds for

Algorithm 14 Generate State Tree

1: **procedure** CREATESTATETREE(Parent Node n, Incident Chain c, Depth d, Stochastic Horizon h^s, Horizon h)
2:     **if** d > horizon h **then**
3:         Return n.util
4:     **else**
5:         $A = \mathit{SelectCandidateActions}(n.state, d, h^s)$
6:         $d \leftarrow d + 1$
7:         Let ChildUtils $\leftarrow \varnothing$
8:         **for** candidate action $a_i \in A$ **do**
9:             Next state $s' \leftarrow \mathit{UpdateState}(n.state, a, c)$
10:             Let cost$_i \leftarrow \mathit{UtilityUpdate}(s', n.cost, d)$
11:             Let $x \leftarrow$ Node(s',cost$_i$,d,t)
12:             ChildUtils $\leftarrow$ ChildUtils $\cup$ $\mathit{CreateStateTree}(x, c, d, h^s, h)$
13:         **end for**
14:     **end if**
15:     Return min(ChildUtils)
16: **end procedure**

Algorithm 15 Generate State Tree

1: **procedure** UTILITYUPDATE(State s, Parent Utility u_p, Depth d, time t)
2:     Return $u_p + (\gamma^t)(t - u_p)/(d + 1))$
3: **end procedure**

the assigned responder to the current incident. Obviously, a lower cost is better, as it corresponds to lower response times. For any given node that was generated by action a from parent node p, we calculate cost as

$$cost = u_p + (\gamma^t)((t - u_p)/(d + 1)) \tag{VIII.3}$$

where u_p is the parent node's cost, γ is the discount factor for future decisions, and t is the time elapsed between taking action a at the parent node and the occurrence of the current node. This is essentially an updated weighted average of the response times to incidents given the dispatch actions.

Once the tree is completed, the cost for each candidate action for the dispatch incident is determined by the cost of the best leaf node in it's sub-tree, as this represents the result of the best sequence of future actions that could be taken given the dispatch action. Finally, the algorithm averages the costs for each dispatch action across the b generated incident chains, and selects the candidate action with the minimum overall cost as the best action in the current state (refer to step 12).

If all the responders are busy when an incident occurs, the incident is placed in a waiting queue.

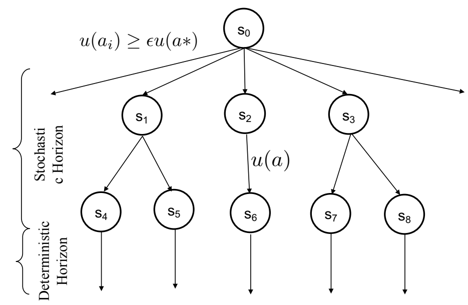


Figure VIII.1: State-Action Tree

As soon as a responder becomes available, it is assigned to the incident at the front of the queue. This continues until the queue is emptied, after which the algorithm returns to using the heuristic policy above.

4 Predicting Environmental Factors

We model the urban area as a set of road segments. For each segment, we assume that the dataset contains an associated set of features, which include data about the number of lanes, length of a segment, vehicular speed at different times and so on. Using this data set and features we learn a function over vehicular speed on a segment, conditional on the set of features using a Long Short-Term Memory Neural Network (LSTM) Hochreiter and Schmidhuber (1997) model [2]. The primary capability of such a framework is to model long-term dependencies and determine the optimal time lag for time series problems, which is especially desirable for traffic prediction in the transportation domain.

5 Performance

5.1 Data and Methodology

Our evaluation uses traffic accident data obtained from the fire and police departments of Nashville, TN, which has a population of approximately 700,000. We trained the generative survival model on 9345 incidents occurring between 1-1-2016 and 2-1-2017, and evaluated the algorithm on 1386 incidents occurring between 2-1-2017 and 4-1-2017. We gathered information about road segments and their geographical locations using real-time traffic data collected from HERE Traffic API (HER, 2018) for Nashville area. The granularity of this dataset lets us access real-time vehicular speed for all segments in Nashville, which is sampled every minute throughout the day.

[2]This part of the work was entirely done by Chinmaya Samal. I present a high-level overview here to make the discussion self-contained, and refer readers to Mukhopadhyay et al. (Mukhopadhyay et al., 2019) for details.

Caching the Router Results: While we recommend using a *router* in real-time using the exact locations of responders and incidents while making decisions, it is not feasible for our experiments. Our preliminary analysis showed that each router request takes approximately 0.2 seconds on average. In order to reduce the query time needed to find vehicular speed between arbitrary locations, we cached travel times between locations for different times of the day, with time discretized every 30 minutes. Our experiments showed that travel times in Nashville do not change significantly at this interval (ranging from 2-7 mph). In order to actualize caching, we used the same grid system as in the incident model, with any location in the city discretized to the centroid of it's grid.

5.2 Experimental Setup

We begin by evaluating the streaming survival model separately by comparing it to a batch-learning approach (Mukhopadhyay et al., 2018). There are two considerations that need to be made during the evaluation -

1. **Decreased Responder Availability**: It is reasonable to assume that the base policy of dispatching the closest responder is correct most of the time and it is only rarely that non-greedy actions are needed. We hypothesize that such situations occur more frequently in practice as the strain on the system is greater: i.e. the incident to responder ratio is higher. This happens since responders attend not only to traffic accidents, but to a variety of other incidents (crimes for example). To take this into account, we ran several experiments with different number of responders: The full Nashville responder count of 26, and then cutting it by a factor of half three times to simulate test-beds with 13, 6, and 3 responders.

2. **Hyper-Parameters**: We performed a hyper-parameter search for each of the test-beds based on the number of stations. The parameters that gave the best response time savings were chosen for each set. We note that each hyper-parameter is important and strongly prescribe that each should be tested and tuned carefully for a new environment and hardware the system is deployed on, as these values may not be optimal for more constrained hardware, different responder distributions, or different cities with other incident arrival models.

5.3 Results and Discussion

5.3.1 Streaming Survival Analysis

We learned the batch model using the entire training data set and then, in the evaluation set, we considered each week as a *stream*, and further split 80% of the stream into a training set and 20%

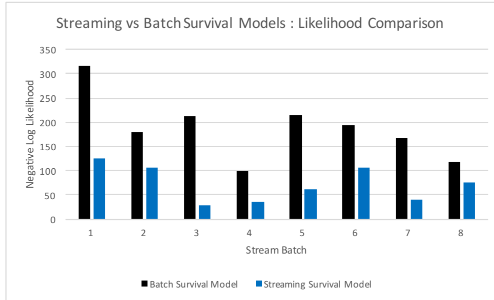


Figure VIII.2: Negative Log Likelihood comparison (lower is better) between Batch and Streaming Survival Models.

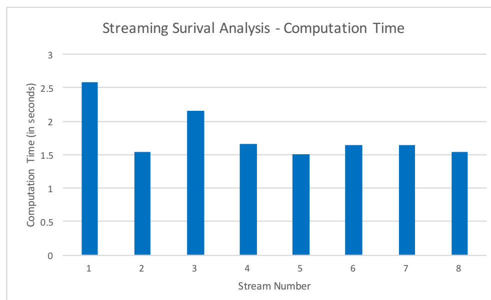
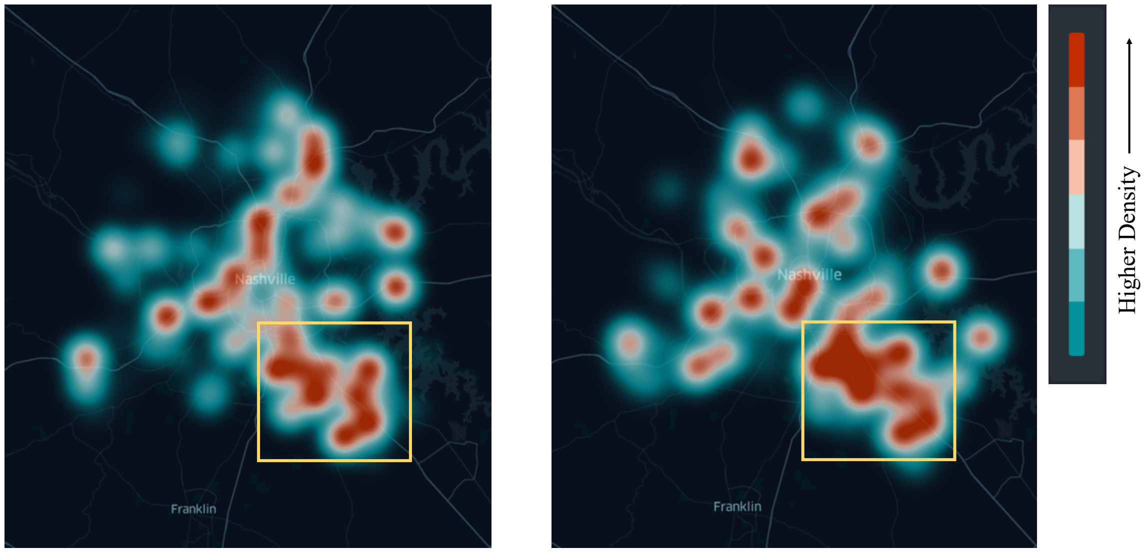


Figure VIII.3: Computational Run-Time for Streaming Survival Analysis.

as the test set. We evaluated the batch model as well as the streaming model on the test set of each of the streams. Note that the batch model has access to all the data in the streams in the form of features; the stream model, on the other hand, gets updated after each data stream is received according to Algorithm 10. We use the *de-facto* standard of comparing likelihoods to evaluate the model, and present the results in in Figure VIII.2. We observe that the streaming model results in a significant increase in likelihood (we plot the negative log likelihoods, hence lower is better) and convincingly outperforms the batch model. We point out a minor caveat - the updates can be used in practice only if the time taken to update the model is small as compared to the latency that emergency responders can afford. To illustrate this, we present the computational run-times of the stream model (for each stream) in Figure VIII.3, and observe that it usually takes less than 2 seconds for an update to run, which justifies the usage of such models in practice.

In order to visually illustrate the benefit of a streaming model, we look at a fabricated example, where we feed the incident prediction models with data that is deviant from standard accident patterns. We show these results as heatmaps in Figure VIII.4. Note that a brighter color corresponds to higher density of incidents. We see that the batch model *weakly* learns the pattern since it has access to the updated dataset only in the form of features; the streaming model on the other hand,



(a) (b)

Figure VIII.4: Batch Model (a) vs Streaming Model (b): These predicted heatmaps demonstrate that the streaming model adjusts more quickly to new incident distributions.

identifies the current trend and predicts higher density of incidents in the concerned region, thereby highlighting the importance of such models in dynamic environments.

Table VIII.1: Performance Summary of System Compared to Base Policy

Number of Stations (Fraction of Nashville Count)	26 (full)	13 (1/2)	6 (1/4)	3 (1/8)
Average Response Time Savings for Incidents Impacted by Policy (seconds)	38.705	2.231	15.917	34.871
Number of Incidents Impacted by Policy	5	14	99	150
Average Computation Time per Incident (seconds)	0.384	0.198	0.350	0.0343

5.3.2 Responder Dispatch

In table VIII.1 we present the results of comparing the tuned algorithms for each stations configuration. We compare our solution against the base policy (sending the nearest responder) using the average response time reduction for incidents impacted by the algorithm (i.e. incidents with different response times than the base policy), the number of incidents impacted, and the average computation time. The first observation is that the computation times are all well within acceptable limits, as they are near the human decision maker's visual reaction times (Taoka, 1989). This demonstrates that the system overcomes the technical challenge of running and updating in real-time, and can integrate into emergency response systems.

We observe that when there is high responder coverage, demonstrated by the experiment with

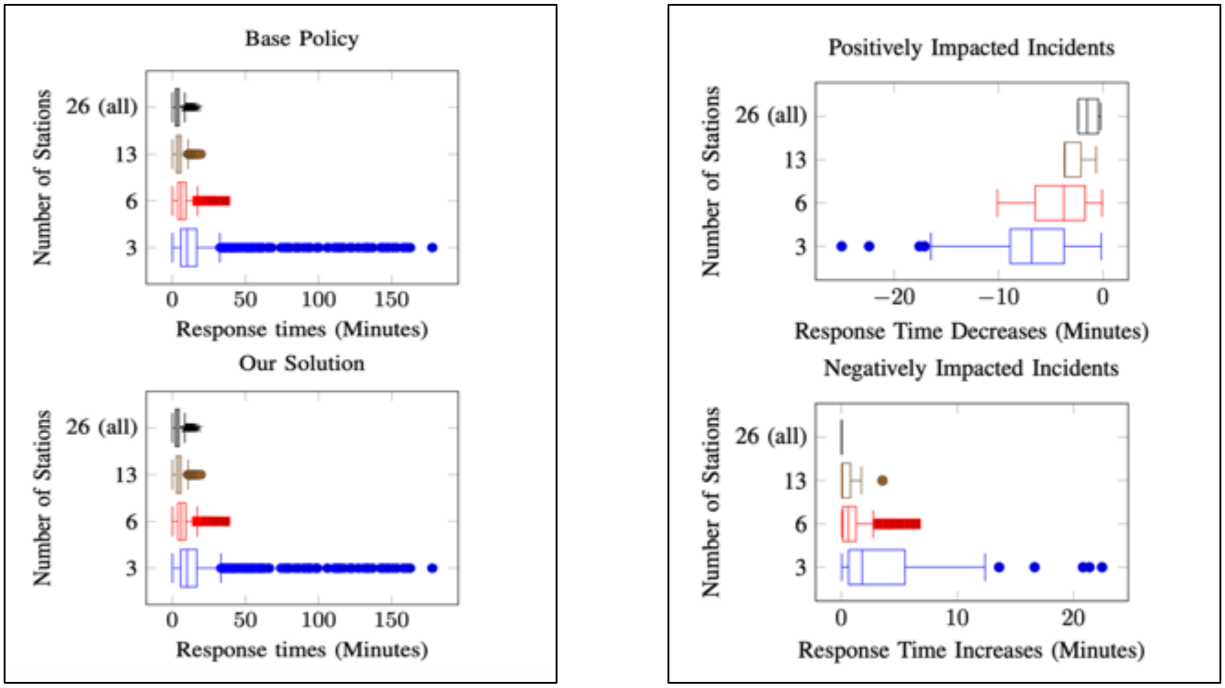


Figure VIII.5: Experimental Results from Online Dispatch Algorithm

Table VIII.2: Detailed Summary of System Compared to Base Policy

Number of Stations	Metric	Greedy Policy (in seconds)	MCTS based Policy (in seconds)
3	Median (Response Time)	603.36	603.21
	Total Positive Impact	21473.69	26704.27
	Average Positive Impact	238.59	445.07
6	Median (Response Time)	369.99	371.28
	Total Positive Impact	4921.53	6497.29
	Average Positive Impact	65.62	240.64
13	Median (Response Time)	256.61	256.70
	Total Positive Impact	445.30	476.53
	Average Positive Impact	31.80	158.84
26	Median (Response Time)	197.63	197.63
	Total Positive Impact	4.31	197.83
	Average Positive Impact	1.07	65.94

26 stations, the baseline policy is nearly always used, with only 5 of the 1386 incidents serviced being impacted by the policy. But as the number of responders decreases, the baseline policy is sub-optimal for an increasing number of incidents, capping at 150 with 3 stations. This shows that the system can respond to changing responder availability, and that it is most useful when the system is strained by high incident demand.

Last, the average time saved for impacted incidents is significant, particularly for the experiments with 26 and 3 stations, as 30 seconds can be the difference between mortality and survival in response situations (Blackwell and Kaufman, 2002). However, these represent average savings, and to dissect the performance of our approach, we present two sets of results; first, we plot the distributions of the response time savings for incidents that benefited from our solution, and response time increases for negatively impacted incidents in figure VIII.5. Comparing the box plots, the negatively impacted response times are more dense near zero compared to the savings. This shows that in general, the algorithm is not making large sacrifices for individual incidents in comparison to the savings generated, which is reinforced by the overall distribution of response times shown in figure VIII.5. The response times for the positively impacted incidents are generally much improved; the median improvement is over 200 seconds for the experiment with 13 stations, for example.

We also present a detailed investigation of the results in Table . For each of the two policies, we present the median savings, total positive impact and the average positive impact across the choice of stations. Although there is not much difference in the median response time achieved by both the policies, the MCTS based policy has more positive impact than the greedy policy. This means that, the incidents which are positively affected by the MCTS based policy provide higher savings than the incidents that are positively affected by the greedy policy. Also, the average savings by the greedy policy decreases as the strain on the system increases, which again points to the fact that a principled approach is significantly better when emergency responders are under stress.

Unfortunately, however, there are some outliers with unacceptably large sacrifices. For example, there is an incident in the experiment with 13 stations that took over 200 additional seconds to respond to compared to the base policy, which significantly increases the potential mortality of that incident if it is severe. This raises the question of integrating severity of incidents into the SMDP model, and we plan to consider the integration of prioritization of incidents in future work.

6 Conclusion

We designed a complete pipeline for the responder dispatch problem. We created an online incident prediction model that can consume streaming data and efficiently update existing incident arrival models. Then, we designed a framework for finding near-optimal decisions of an SMDP by using Monte-Carlo Tree Search, that bridges an important gap in literature by making such models computationally tractable. To aid the decision-making algorithm, we designed a Recurrent Neural Network architecture to learn and predict traffic conditions in urban areas. Our experiments showed significant improvements over prior work and existing strategies in both incident prediction and responder dispatch.

CHAPTER IX

Research to Practice

1 Introduction

There have been two broad goals of this thesis. First, and most importantly, we have sought to better the algorithmic state-of-the-art in robust incidence prediction, resource allocation and dynamic dispatch. However, an important yardstick to judge the efficacy of such research is to evaluate its impact in practice. This brings us to our second goal - ensuring that such algorithmic approaches are properly implemented, packaged and handed over to first responders and police departments who can use it to help the society. In this chapter, we will briefly go over a tool that we developed [1] as part of this project, and discuss challenges and lessons learned as part of the process. Also, we will describe how the tool is actually being used in practice.

2 *VData*: A one-stop tool for emergency responders

We now describe a specific tool-chain that was created to aid the Nashville Police and Fire departments in emergency response. This was always one of the goals of this project, and the specific requirements for the tool were set in conjunction with the necessary inputs from the said government bodies.

We created an open-source web-based platform that acts as a one-stop complete solution for incident response. The tool facilitates low-latency bi-directional communication between clients (emergency response stations) and a central server. A high-level software architecture for the tool is shown in Fig. IX.1. First, each emergency response station acts as a client and can access real-time updates about incidents, current positions of emergency responders as well analytics on historical data instantly. Secondly, as soon as an incident happens, all clients are notified of the location of the incident, as well as an optimal action based on the dispatch approach introduced in chapter VII. Finally, users can access an exploratory mode that aids long-term decision making. We briefly describe the functionality of our tool and present screen captures.

There are four major modes in the tool -

1. Historical – The mode enables user to visualize historical data as heat map/ incident scatter plots on a geographical map of Nashville.

[1] The development of the tool was done in collaboration with Zilin Wang, who did most of the work for the front-end of the tool. During deployment, the work was done in collaboration with Geoffrey Pettet

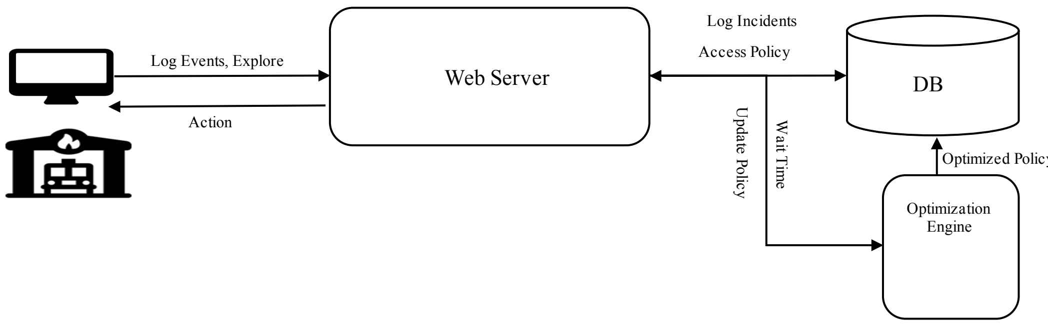


Figure IX.1: VData : Architecture

2. Prediction – Enables the user to predict incidents for arbitrary future dates according the incident type.

3. Exploratory – Lets the user add a new fire station, and calculate its effect on the response time.

4. Dispatching – Helps the user in plotting pending incidents that need to be serviced, and provides dispatch decision based on a responder dispatch algorithm.

At the back end, the tool has three primary background jobs – data ingestion to the database layer used by the dashboard, update of the prediction algorithm with streaming data and overall monitoring of the tool.

The tool provides the ability to each client to visualize historical data in the form of both markers and heat maps, as well as analytics based on such incidents, thus helping in providing an immediate visual summary of incidents. Users can also instantly shift to a *future* mode, and predict incidents based on Eq.VII.1 for specific dates or date ranges, which serves as an important mechanism for resource planning and budget allocations. Finally, when an incident happens, the server pushes a notification to each client about the location of the incident as a marker on the map. It also highlights which emergency responder should be sent to the site of the incident. This feature provides crucial assistance to the emergency responders thus aiding real-time decision making. The tool also provides an exploratory mode for making long-term decisions. It provides the ability to users to add a new depot at a specific location, allocate responders to it and understand how much the expected wait time changes upon the addition of such a depot. This helps organizations strategically during expansions or relocating depots and/or responders.

3 Deployment

The tool has been handed over to the Nashville Fire Department. There are still changes that need to be done to make the tool customized for the Metropolitan Nashville Police Department.

4 Discussion

An important responsibility that comes with working on a tool for emergency response is passing the lessons learned to the community, such that future researchers and developers can learn from our experience. In our opinion, the following considerations are imperative while such a tool is planned or developed.

1. Domain Expertise in research methodology - The most crucial consideration for researchers should be an acknowledgement of lack of domain expertise on their part. Despite having done significant research on the problem domain of emergency response, our work was greatly enhanced by suggestions made by both the fire and police departments of Nasvhille.

2. Feedback from the field - Ultimately, such tools of emergency response would be used by first-responders in the field, and often times, they understand the requirements and the adequate features significantly better than others. It is important that first responders are involved to set initial requirements of such tools, and subsequently used for testing different versions, thereby improving the tool gradually.

3. Domain specific needs - In our experience, different emergency responders have completely different needs and methodology for allocating and dispatching responders. It is crucial that such needs are considered early on, and tools are created that suit the specific needs of the organization.

Through this thesis, we have tried to create a holistic pipeline for addressing the problem of lack of efficient emergency response in urban areas. While we have looked at several problems and designed novel algorithmic approaches to tackle them, much needs to be done. Emergency response needs principled approaches, and part of the initiative has to be taken by first responders to accept algorithmic work being done by technical research groups. At the same time, one should tread carefully while designing such algorithms, since emergency response essentially deals with life and death scenarios. A major consideration for research groups should be look at this problem in its entirety, if possible. While the atomic problems provide ample technical challenges on their own, the true impact of such approaches can be made by creating and deploying pipelines that can be deployed in practice and aid emergency responders.

CHAPTER X

Conclusion

Over the course of this doctoral thesis, we have managed to identify, model and address several challenges in the domain of emergency response. To begin with, we have identified three major sub-problems that constitute the broader problem of emergency response - incident forecasting, responder allocation and responder dispatch. We have looked at these three problems separately, as well as studied and evaluated how they work in conjunction with each other.

We started by creating a continuous-time model for forecasting incidents in space and time, that can accommodate arbitrary covariates. We also created principled approaches to estimate the severity of such incidents and learn the spatial granularity at which such models should be learned. While we focused on various categories of incidents, forecasting crimes presented particularly interesting challenges. Unlike other incidents (that are labeled as emergencies), crimes are usually deliberate acts; as a consequence, criminals have a direct incentives to evade forecasting models by altering their behavior. In order to take this into account, we systematically modeled behavior manipulation by criminals in response to crime forecasting models, and created forecasting algorithms that are robust against such manipulation.

The second problem that we tackled was that of allocating emergency responders in anticipation of incidents. We first clearly separated this problem for police and ambulances. Police vehicles, by the virtue of their presence affect the distribution of crimes, whereas ambulances lack the ability to affect the distribution of accidents. As a result, we modeled the police stationing problem as a two-stage optimization problem, and used iterative application of Bender's decomposition to solve the problem. Modeling the placement of ambulances posed a different challenge. Ambulances often times work under legal constraints with respect to waiting times, and their response strategy often varies with the reported severity of incidents. Integrating such constraints in the stationing problem results in the formation of highly non-linear and non-convex optimization problems. To tackle such problems, we crafted specific heuristics based on Greedy-Random Adaptive Search (GRASP).

Finally, we looked at the problem of dispatching first responders as and when incidents occurred. Such a problem typically evolves in a semi-Markovian environment with unknown transition probabilities. We created a novel approach to solve for optimal policies in such problems by learning the transition functions on the fly. However, for any reasonable sized problem with a dynamic environment, directly solving a large-scale semi-Markovian decision process provides limited benefits. In order to tackle this, we created online approaches to incident forecasting and used Monte-Carlo methods to efficiently find near-optimal actions for the decision problem of dispatching responders. All our work was evaluated using real-world data from the Nashville, TN.

Although we have extensively looked at the problem of emergency response, there are plenty of opportunities as well as need for future work. Two directions of future work are particularly interesting that we are already pursuing. In robust incident prediction, a possible shortcoming of our approach is that we assume perfect rationality from criminals when they respond to specific models of incident forecasting. This is however, not entirely valid in practice. An interesting direction of future work is to include well studied models of human decision-making into our model and create robust algorithmic approaches to incident prediction. The second avenue of work comes from the problem of responder dispatch. Modern ambulances today are equipped with computational capabilities, and leveraging such computing power could be extremely beneficial. Specifically, in the disaster situations, agents often fail to communicate within themselves or central authorities, and decentralizing the planning problem could avoid the problem with a single point of failure. Further, decentralized problem also promises to be significantly more efficient, which could prove to be very important in life and death scenarios.

BIBLIOGRAPHY

(2018). Here api. https://developer.here.com/. Accessed: 2018-11-17.

Ackaah, W. and Salifu, M. (2011). Crash prediction model for two-lane rural highways in the ashanti region of ghana. *IATSS research*, 35(1):34–40.

Amendola, K. L., Weisburd, D., Hamilton, E., Jones, G., Slipka, M., Heitmann, A., Shane, J., Ortiz, C., and Tarkghen, E. (2011). The shift length experiment: What we know about 8-, 10-, and 12-hour shifts in policing. *Police Foundation. Advancing Plicing Through Innovation and Science*.

Balke, K., Chaudhary, N., Chu, C.-L., Kuchangi, S., Nelson, P., Songchitruksa, P., Swaroop, D., and Tyagi, V. (2005). Dynamic traffic flow modeling for incident detection and short-term congestion prediction: Year 1 progress report. Technical report.

Barreno, M., Nelson, B., Sears, R., Joseph, A. D., and Tygar, J. D. (2006). Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pages 16–25. ACM.

Berman, O. and Larson, R. C. (1982). The median problem with congestion. *Computers & Operations Research*, 9(2):119–126.

Berman, O., Larson, R. C., and Chiu, S. S. (1985). Optimal server location on a network operating as an m/g/1 queue. *Operations research*, 33(4):746–771.

Bertsimas, D. and Tsitsiklis, J. N. (1997). *Linear Optimization*. Athena Scientific, 3rd edition.

Blackwell, T. H. and Kaufman, J. S. (2002). Response time effectiveness: comparison of response time and survival in an urban emergency medical services system. *Academic Emergency Medicine*, 9(4):288–295.

Böhning, D. (1992). Multinomial logistic regression algorithm. *Annals of the Institute of Statistical Mathematics*, 44(1):197–200.

Bonneson, J. A. and McCoy, P. T. (1993). Estimation of safety at two-way stop-controlled intersections on rural highways. *Transportation Research Record*, (1401).

Brandeau, M. L. and Chiu, S. S. (1992). A center location problem with congestion. *Annals of operations research*, 40(1):17–32.

Brown, M., Sinha, A., Schlenker, A., and Tambe, M. (2016). One size does not fit all: A game-theoretic approach for dynamically and effectively screening for threats. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Chakravorty, S. (1995). Identifying crime clusters: The spatial principles. *Middle States Geographer*, 28:53–58.

Chrobak, M., Karloof, H., Payne, T., and Vishwnathan, S. (1991). New ressults on server problems. *SIAM Journal on Discrete Mathematics*, 4(2):172–181.

Church, R. and Velle, C. R. (1974). The maximal covering location problem. *Papers in regional science*, 32(1):101–118.

Clarke, R. V. (2013). Affect and the reasoning criminal: Past and future. In *Affect and cognition in criminal decision making*, pages 38–59. Routledge.

Cobham, A. (1954). Priority assignment in waiting line problems. *Journal of the Operations Research Society of America*, 2(1):70–76.

Cohn, E. G. (1990). Weather and crime. *British journal of criminology*, 30(1):51–64.

Cox, D. R. and Oakes, D. (1984). *Analysis of survival data*, volume 21. CRC Press.

Davis, R. (2005). Six minutes to live or die. *USA Today*.

Eaton, D. J., Daskin, M. S., Simmons, D., Bulloch, B., and Jansma, G. (1985). Determining emergency medical service vehicle deployment in austin, texas. *Interfaces*, 15(1):96–108.

Fang, F., Nguyen, T. H., Pickles, R., Lam, W. Y., Clements, G. R., An, B., Singh, A., Tambe, M., and Lemieux, A. (2016). Deploying paws: Field optimization of the protection assistant for wildlife security. In *Twenty-Eighth IAAI Conference*.

Farrell, G. and Pease, K. (2014). *Prediction and Crime Clusters*, pages 3862–3871. Springer New York, New York, NY.

Felder, S. and Brinkmann, H. (2002). Spatial allocation of emergency medical services: minimising the death rate or providing equal access? *Regional Science and Urban Economics*, 32(1):27–45.

Felson, M. and Poulsen, E. (2003). Simple indicators of crime by time of day. *International Journal of Forecasting*, 19(4):595–601.

Frantzeskakis, J., Assimakopoulos, V., and Kindinis, G. (1994). Interurban accident prediction by administrative area application in greece. *ITE journal*, 64(1):35–42.

Gholami, S., Mc Carthy, S., Dilkina, B., Plumptre, A., Tambe, M., Driciru, M., Wanyama, F., Rwetsiba, A., Nsubaga, M., Mabonga, J., et al. (2018). Adversary models account for imperfect crime data: Forecasting and planning against real-world poachers. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 823–831. International Foundation for Autonomous Agents and Multiagent Systems.

Gorr, W., Olligschlaeger, A., and Thompson, Y. (2003). Short-term forecasting of crime. *International Journal of Forecasting*, 19(4):579–594.

Guha, S. and Khuller, S. (1999). Greedy strikes back: Improved facility location algorithm. *Journal of Algorithms*, 31(1):228–248.

Hakimi, S. L. (1964). Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations research*, 12(3):450–459.

Haklay, M. M. and Weber, P. (2008). Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4):12–18.

Harchol-Balter, M., Osogami, T., Scheller-Wolf, A., and Wierman, A. (2005). Multi-server queueing systems with multiple priority classes. *Queueing Systems*, 51(3-4):331–360.

Haskell, W., Kar, D., Fang, F., Tambe, M., Cheung, S., and Denicola, E. (2014). Robust protection of fisheries with compass. In *Twenty-Sixth IAAI Conference*.

Hindelang, M. J., Gottfredson, M. R., and Garofalo, J. (1978). *Victims of personal crime: An empirical foundation for a theory of personal victimization*. Ballinger Cambridge, MA.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hope, T. (1994). Problem-oriented policing and drug market locations: Three case studies. *Crime prevention studies*, 2(1):5–32.

Hu, Q. and Yue, W. (2007). *Markov decision processes with their applications*, volume 14. Springer Science & Business Media.

Huang, L., Joseph, A. D., Nelson, B., Rubinstein, B. I., and Tygar, J. (2011). Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58. ACM.

Ivaha, C., Al-Madfai, H., Higgs, G., and Ware, J. A. (2007). The dynamic spatial disaggregation approach: A spatio-temporal modelling of crime. In *World Congress on Engineering*, pages 961–966.

Johnson, M. P., Fang, F., and Tambe, M. (2012). Patrol strategies to maximize pristine forest area. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254.

Jones, E., Oliphant, T., Peterson, P., et al. (2001–). SciPy: Open source scientific tools for Python.

Kearns, M., Mansour, Y., and Ng, A. Y. (2002). A sparse sampling algorithm for near-optimal planning in large markov decision processes. *Machine learning*, 49(2):193–208.

Kendall, D. G. (1953). Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded markov chain. *The Annals of Mathematical Statistics*, pages 338–354.

Kennedy, L. W., Caplan, J. M., and Piza, E. (2011). Risk clusters, hotspots, and spatial intelligence: risk terrain modeling as an algorithm for police resource allocation strategies. *Journal of Quantitative Criminology*, 27(3):339–362.

Kiattikomol, V. (2005). Freeway crash prediction models for long-range urban transportation planning.

Kleemans, E. R. (2001). Repeat burglary victimisation: results of empirical research in the netherlands. *Crime prevention studies*, 12:53–68.

Koper, C. S. (1995). Just enough police presence: Reducing crime and disorderly behavior by optimizing patrol time in crime hot spots. *Justice Quarterly*, 12(4):649–672.

Landau, S. F. and Fridman, D. (1993). The seasonality of violent crime: the case of robbery and homicide in israel. *Journal of research in crime and delinquency*, 30(2):163–191.

Lauritsen, J. L. and White, N. (2014). *Seasonal Patterns in Criminal Victimization Trends*, volume 245959. US DOJ, Office of Justice Program, Bureau of Justice Statistics.

Levine, N. et al. (2004). Crimestat III: a spatial statistics program for the analysis of crime incident locations (version 3.0). *Houston (TX): Ned Levine & Associates/Washington, DC: National Institute of Justice*.

Li, S. (2011). A 1.488 approximation algorithm for the uncapacitated facility location problem. *Automata, Languages and Programming*, 6756:77–88.

Li, X., Zhao, Z., Zhu, X., and Wyatt, T. (2011). Covering models and optimization techniques for emergency response facility location and planning: a review. *Mathematical Methods of Operations Research*, 74(3):281–310.

Maher, M. J. and Summersgill, I. (1996). A comprehensive methodology for the fitting of predictive accident models. *Accident Analysis & Prevention*, 28(3):281–296.

Majzoubi, F. (2014). *A Mathematical Programming Approach for Dispatching and Relocating EMS Vehicles*. PhD thesis, University of Louisville.

Miller Jr, R. G. (2011). *Survival analysis*, volume 66. John Wiley & Sons.

Mohler, G. O., Short, M. B., Brantingham, P. J., Schoenberg, F. P., and Tita, G. E. (2011). Self-exciting point process modeling of crime. *Journal of the American Statistical Association*, 106(493):100–108.

Mukhopadhyay, A., Pettet, G., Samal, C., Dubey, A., and Vorobeychik, Y. (2019). An online decision-theoretic pipeline for responder dispatch. In *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, ICCPS '19, pages 185–196, New York, NY, USA. ACM.

Mukhopadhyay, A., Vorobeychik, Y., Dubey, A., and Biswas, G. (2017). Prioritized allocation of emergency responders based on a continuous-time incident prediction model. In *International Conference on Autonomous Agents and MultiAgent Systems*, pages 168–177. International Foundation for Autonomous Agents and Multiagent Systems.

Mukhopadhyay, A., Wang, Z., and Vorobeychik, Y. (2018). A decision theoretic framework for emergency responder dispatch. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS 2018, Stockholm, Sweden, July 10-15, 2018*, pages 588–596.

Mukhopadhyay, A., Zhang, C., Vorobeychik, Y., Tambe, M., Pence, K., and Speer, P. (2016a). Optimal allocation of police patrol resources using a continuous-time crime model. In *Conference on Decision and Game Theory for Security*.

Mukhopadhyay, A., Zhang, C., Vorobeychik, Y., Tambe, M., Pence, K., and Speer, P. (2016b). Optimal allocation of police patrol resources using a continuous-time crime model. In *International Conference on Decision and Game Theory for Security*, pages 139–158. Springer.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38.

Murray, A. T., McGuffog, I., Western, J. S., and Mullins, P. (2001). Exploratory spatial data analysis techniques for examining urban crime implications for evaluating treatment. *British Journal of criminology*, 41(2):309–329.

Nolan, J. J. (2004). Establishing the statistical relationship between population size and ucr crime rate: Its impact and implications. *Journal of Criminal Justice*, 32(6):547–555.

Péret, L. and Garcia, F. (2004). On-line search for solving markov decision processes via heuristic sampling. In *Proceedings of the 16th European Conference on Artificial Intelligence*, pages 530–534. IOS Press.

Poch, M. and Mannering, F. (1996). Negative binomial analysis of intersection-accident frequencies. *Journal of Transportation Engineering*, 122(2):105–113.

Private Communication (2018). Police Department of a major metropolitan city (anonymized for blind review).

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Qi, Y., Smith, B. L., and Guo, J. (2007). Freeway accident likelihood prediction using a panel data analysis approach. *Journal of transportation engineering*, 133(3):149–156.

Reppetto, T. A. (1976). Crime prevention and the displacement phenomenon. *Crime & Delinquency*, 22(2):166–177.

Resende, P. T. V. and Benekohal, R. F. (1997). Development of volume-to-capacity based accident prediction models. In *Traffic Congestion and Traffic Safety in the 21st Century: Challenges, Innovations, and Opportunities*.

Sayed, T. and Rodriguez, F. (1999). Accident prediction models for urban unsignalized intersections in british columbia. *Transportation Research Record: Journal of the Transportation Research Board*, (1665):93–99.

Shankar, V., Mannering, F., and Barfield, W. (1995). Effect of roadway geometrics and environmental factors on rural freeway accident frequencies. *Accident Analysis & Prevention*, 27(3):371–389.

Short, M. B., D'Orsogna, M. R., Pasour, V. B., Tita, G., Brantingham, P. J., Bertozzi, A. L. ., and Chayes, L. B. (2008a). A statistical model of criminal behavior. *Mathematical Models and Methods in Applied Sciences*, pages 1249–1267.

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

Silva, F. and Serra, D. (2008). Locating emergency services with different priorities: the priority queuing covering location problem. *Journal of the Operational Research Society*, 59(9):1229–1238.

Songchitruksa, P. and Balke, K. (2006). Assessing weather, environment, and loop data for real-time freeway incident prediction. *Transportation Research Record: Journal of the Transportation Research Board*, (1959):105–113.

Speer, P. W., Gorman, D. M., Labouvie, E. W., and Ontkush, M. J. (1998). Violent crime and alcohol availability: Relationships in an urban community. *Journal of Public Health Policy*, 19(3):303–318.

Taoka, G. T. (1989). Brake reaction times of unalerted drivers. *ITE Journal*, 59(3):19–21.

Toomey, T. L., Erickson, D. J., Carlin, B. P., Quick, H. S., Harwood, E. M., Lenk, K. M., and Ecklund, A. M. (2012). Is the density of alcohol establishments related to nonviolent crime? *J Stud Alcohol Drugs*, 73(1):21–25.

Toro-DíAz, H., Mayorga, M. E., Chanta, S., and Mclay, L. A. (2013). Joint location and dispatching decisions for emergency medical services. *Computers & Industrial Engineering*, 64(4):917–928.

Tsai, J., Kiekintveld, C., Ordonez, F., Tambe, M., and Rathi, S. (2009). Iris-a tool for strategic security allocation in transportation networks.

Von Stengel, B. and Zamir, S. (2004). Leadership with commitment to mixed strategies. Technical report, Citeseer.

Vorobeychik, Y. (2016). Adversarial ai. In *IJCAI*, pages 4094–4099.

Zhang, C., Bucarey, V., Mukhopadhyay, A., Sinha, A., Qian, Y., Vorobeychik, Y., and Tambe, M. (2016). Using abstractions to solve opportunistic crime security games at scale. In *International Conference on Autonomous Agents and Multiagent Systems*.

Zhang, C., Sinha, A., and Tambe, M. (2015). Keeping pace with criminals: Designing patrol allocation against adaptive opportunistic criminals. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 1351–1359.

Zhanli, M., Yi, Z., Bozhong, Y., and Lei, Z. (2010). Forecast of fire accidents based on grey-markov model [j]. *Engineering Sciences*, 1:019.

Zografos, K. G., Androutsopoulos, K. N., and Vasilakis, G. M. (2002). A real-time decision support system for roadway network incident response logistics. *Transportation Research Part C: Emerging Technologies*, 10(1):1–18.

Zografos, K. G., Nathanail, T., and Michalopoulos, P. (1993). Analytical framework for minimizing freeway-incident response time. *Journal of Transportation Engineering*, 119(4):535–549.