Sequential Decision Making in Crisis: Mitigating Risk in Marginalized Communities with Stochastic Optimization

by

Kati Moug

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in The University of Michigan
2023

Doctoral Committee:

Associate Professor Siqian Shen, Chair
Professor Brian Denton
Associate Professor Cong Shi
Professor Lei Ying

Kati Moug

klmoug@umich.edu

ORCID iD: 0000-0002-5032-931X

© Kati Moug 2023

ACKNOWLEDGEMENTS

I would like to thank everyone who helped make this dissertation possible. Above all, I would like to thank my advisor, Dr. Siqian Shen, for giving me the opportunity to explore social problems through the lens of operations research. Your guidance, expertise, and wisdom have brought me to where I am today.

I also want to thank my committee members, Dr. Brian Denton, Dr. Cong Shi, and Dr. Lei Ying, for their insight and thoughtful comments. As I look forward to the future, I want to thank Dr. Shen, Dr. Denton, and my teaching advisor, Dr. Lawrence Seiford, for their career advice and mentorship. I also want to thank the professors in IOE that have taught me over the years for their passion and expertise.

I also am thankful to my labmates, especially Dr. Huiwen Jia, for their support in my development as a researcher. I am grateful to my cohort at IOE, who made learning joyful, and to Grads Out in STEM, who provided support and camaraderie. A special thanks to Ypsilanti Rugby, who made the Ann Arbor area my home.

Finally, I want to thank my family. Thanks to my sister, Mackenzie Moug, for completing a half-marathon and always believing I could complete this one. Thanks to my father, Patrick Moug, for showing me humans can be "both" rather than "either." I wouldn't be a mathematician-sociologist without you. And thanks to my mother, Kim Wilkins-Moug, for being the first scientist in my life. I march to the beat of my own drum because you do.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS ii

LIST OF FIGURES vi

LIST OF TABLES viii

ABSTRACT x

CHAPTER

I. Introduction 1

1.1 Volunteer Recruitment and Operations for Evacuation Ridesharing 3
1.2 De-Densification of Isolated Facilities for Pandemic Containment 5
1.3 Fake News Mitigation with Stochastic Network Interdiction . 8
1.4 Structure of the Dissertation 9

II. A Shared Mobility Based Framework for Evacuation Planning and Operations under Forecast Uncertainty 11

2.1 Introductory Remarks 11
2.1.1 Prior work on evacuation via ridesharing 13
2.1.2 Contributions of the chapter 14
2.1.3 Structure of the chapter 15
2.2 Mathematical Models 15
2.2.1 Notation, parameters and basic assumptions 16
2.2.2 Expectation-based formulation 19
2.2.3 Chance-constrained program 25
2.2.4 CVaR approximation of chance-constrained program 29
2.3 Solution Algorithm 31
2.4 Generation and Parameter Settings of Test Instances 35
2.4.1 Scenario generation method overview 36
2.4.2 Parameter settings 37

2.5 Computational Studies 43
2.5.1 Comparison of (Exp-SAA) and heuristic for $\gamma = 1$. 45
2.5.2 Comparison of different stochastic optimization models 50
2.6 Concluding Remarks 53

III. The Costs of Overcrowding (and Release): Strategic Discharges for Isolated Facilities During Epidemiological Outbreaks 56

3.1 Introductory Remarks 56
3.1.1 Methodological Contributions 56
3.1.2 Broader Contributions 57
3.2 Literature Review 58
3.2.1 Epidemiological Models 59
3.2.2 Optimal Control in Epidemiology 60
3.2.3 Dynamic Control and Uncertain Parameters 61
3.3 Strategic Releases and the De-Densification Effect 63
3.4 Decision Making Models 66
3.4.1 Markov Decision Processes 67
3.4.2 Model Parameters 69
3.4.3 Generative Markov Decision Processes 71
3.5 Solution Methods 71
3.5.1 Sparse Sampling-Based Methods 71
3.5.2 Q Learning with Linear Function Approximation . . 78
3.6 Benchmark De-Densification Test Instances 83
3.6.1 Epidemiological Outbreak Test Instances 83
3.6.2 Epidemiological Parameters 86
3.6.3 Action Space Definition 87
3.6.4 Contact Rate Assumptions 88
3.6.5 Multiobjective Parameters 89
3.7 Open-Source Generic Markov Simulation Code 90
3.8 Computational Experiments 92
3.8.1 Experimental Setup 93
3.8.2 Comparison Across Methods 94
3.8.3 Comparison Across Policy Weights 96
3.9 Concluding Remarks 98

IV. Stochastic Network Interdiction-Based Mitigation of Fake News 100

4.1 Introductory Remarks 100
4.2 Mathematical Model 102
4.2.1 Social Media Network Parameters 102
4.2.2 Stochastic Network Interdiction Formulation 106
4.2.3 Adaptation for Scalability 109
4.3 Path Networks with Stochastic System Interdiction 112

4.3.1 Two Target Nodes 114
4.3.2 ϵ-Optimal Approximation Algorithm for Path Networks 116
4.4 Numerical Studies 118
4.4.1 Path Networks with Two Target Nodes 118
4.4.2 ϵ-Optimal Algorithm for Path Networks 119
4.4.3 Scalable Mixed-Integer Program for Networks with Community Structure 120
4.5 Concluding Remarks 129

V. Conclusion 131

BIBLIOGRAPHY 134

LIST OF FIGURES

Figure

2.1 An example of (a) geographical network $\mathcal{L}$ with 6 locations, labeled with travel time t_e, $\forall e \in \mathcal{A}'$ and (b) its spatiotemporal network $\mathcal{G}$ with $T = 5$, demand realization $\mathbf{a}(\omega)$ and vehicle capacities $\mathcal{Q}_v$ for $v \in V$. 19

2.2 Illustration of regions and demand hot zones in Charleston County. 37

2.3 Hourly traffic counts for September 1 to September 13, 2018 on Highway I-26 in Charleston County. 39

2.4 Spatial distributions of expected evacuation demand and number of volunteers in Regions A and B. 43

2.5 Spatial distributions of expected evacuation demand and number of volunteers in Regions C and D. 44

2.6 Spatial distributions of expected evacuation demand and number of volunteers in Region E. 44

2.7 Total number of volunteers hired for Region D for six test periods. . 48

2.8 In-sample performance for three methods for Region C, Period 5. . . 52

2.9 Chosen solution performance for three methods for Regions C and E, Period 5. 54

3.1 Example of an MDP transition during epoch $t = 0, \ldots, T-1$, with current state j_t, action a_t chosen by policy $\hat{\pi}(j_t)$, and transition to next state j_{t+1} governed by probability distribution $P_{j_t \cdot}(a_t)$. Cost $\rho(j_t, a_t)$ is accrued immediately. 68

3.2 (a) Priority for prisoner release. Darker boxes are higher priority. (b) Available actions in each epoch. For each action, all currently susceptible prisoners in the shaded categories are released. 87

3.3 Number of 1000 out-of-sample trajectory policies given by cluster method of each type, for polynomial contact rate assumption and three priority weights. 97

3.4 Mean cost by cost type over planning horizon for different priority weights under polynomial contact rate assumption for out-of-sample tests using cluster algorithm policy. 97

3.5 Total mean discounted cost by cost type for different priority weights under polynomial contact rate assumption for cluster algorithm out-of-sample tests. 98
4.1 Graph $G = (N, A)$ with flow variable y_ω edge labels for scenario $\omega \in \Omega$. Edges with no labels have 0 flow. 103
4.2 Graph $G = (N, A)$ with interdiction. System-initiated interdiction is represented by a large X, and user-initiated interdiction by a small X. The graph includes flow variable y_ω edge labels for scenario $\omega \in \Omega$. Edges with no labels have 0 flow. 105
4.3 Graph $\mathcal{P}_n$. 112
4.4 The threshold multiplier in the delayed system-initiated interdiction threshold, $1 + 1/q_2^\alpha$, as a function of user-initiated interdiction probability p_2^α. 114
4.5 Path network $\mathcal{P}_2$ used in our numerical studies. 118
4.6 Community graph with $|S^c| = 1$ and stochastic users. 125
4.7 Community graph with $|S^c| = 1$ and both stochastic users and system. 127
4.8 Community graph with $|S^c| = 1$ and stochastic system. 128

LIST OF TABLES

Table

2.1 Overview and comparison of different risk measures and their corresponding formulations for evacuation planning. 14
2.2 A summary of notation and definitions. 17
2.3 Percentage of 65+ population requesting ridesharing-based evacuation service in census tract 39
2.4 Temporal distribution of evacuees at Site 31 during different hours on September 11, 2018. 41
2.5 Total expected demand for each region and period. 42
2.6 Gurobi parameter settings for solving (Exp-SAA) versus Algorithm 1 comparison 47
2.7 Problem instance sizes during the busiest period for in-sample tests for (Exp-SAA), including total expected network period demand, $\mu_2^{\mathcal{N}'}$. 47
2.8 Problem instance sizes for smallest and largest networks, Region A and D, for in-sample tests for (Exp-SAA), including total expected network period demand, $\mu_\zeta^{\mathcal{N}'}$. 47
2.9 In-sample computational runtime (in second) and out-of-sample objective values for (Exp-SAA) and Algorithm 1 for the busiest period, Period 2. 49
2.10 In-sample computational runtime (in second) and out-of-sample objective values for (Exp-SAA) and Algorithm 1 for all tests. 49
2.11 Out-of-sample solution performance of expectation-based formulation and Algorithm 1 for the busiest period, Period 2. 50
2.12 Out-of-sample solution performance of expectation-based formulation and Algorithm 1 for all tests. 50
2.13 Problem instance sizes for three methods' in-sample tests, including total expected network period demand, $\mu_5^{\mathcal{N}'}$. 51
3.1 Summary of Notation and Key Parameter 67
3.2 The table below shows an example state $i \in \Omega$ with virus states $\mathcal{V} = \{S, I, R\}$, health risk levels $\mathcal{H} = \{h_1, h_2\}$, and release categories $\mathcal{R} = \{r_1, r_2, r^*\}$. This is the initial state in our numerical studies. . . 69

3.3 Example action $a \in \mathcal{A}(i)$ for state $i \in \Omega$, given in Table 3.2. This corresponds to the high health, low recidivism action in our numerical studies. 70
3.4 Example next state $j \in \Omega$ with virus states $\mathcal{V} = \{S, I, R\}$, health risk levels $\mathcal{H} = \{h_1, h_2\}$, and release categories $\mathcal{R} = \{r_1, r_2, r^*\}$, taken from a static benchmark simulation trajectory. 71
3.5 Epidemiological Parameters 86
3.6 Categories that must have currently susceptible individuals for each action to be taken. 88
3.7 Quadratic coefficients for de-densification functions. 89
3.8 Reward parameters. 90
3.9 Weight parameters. 90
3.10 ST-SpSa parameters. 93
3.11 Q learning with linear function approximation hyperparameters tested. 94
3.12 Percentage improvement in total mean discounted reward for different methods over static benchmark, for different contact assumptions and priority weights. 95
3.13 Probability of epidemic dying out after one epoch for each action in static benchmark trajectories and sparse sampling lookahead trees, with and without clusters. 95
3.14 Runtime (hours) for each computational method. 96
4.1 Interdicted edge and expected number of target nodes reached for $\mathcal{P}_2$ test cases. 119
4.2 Search space size and computational runtime for $\mathcal{P}_{100}$ test cases for ϵ-optimal algorithm, compared to naive benchmark. 120
4.3 Random Behavior Parameters 121
4.4 Final Objective Value by Test Instance 122
4.5 Size of Communities in Interdicted Edges by Test Instance 124
4.6 Characteristics of Interdicted Edges by Test Instance 126

ABSTRACT

In this dissertation, we develop operations research models and solution approaches to three community-based problems: volunteer recruitment for shared-mobility-based evacuation services, de-densification of crowded facilities during epidemiological outbreaks, and fake news and hate speech mitigation in social media networks. These models incorporate sequential decision making – resource planning before evacuation needs are fully known, dynamic decisions made through the course of an outbreak, and the interplay between social media network regulators and users. They also involve uncertainty – stochastic models of a carless population's need for evacuation services, of epidemiological dynamics, and of individual willingness to believe and share fake news.

In Chapter II, we propose a ridesharing-based evacuation program that recruits volunteer drivers before a disaster strikes, and then matches volunteer drivers with evacuees once demand is realized. We optimize resource planning and evacuation operations under uncertain spatiotemporal demand, and construct a two-stage stochastic mixed-integer program to ensure high demand fulfillment rates. We consider three formulations to improve the number of evacuees served, by minimizing an expected penalty cost, imposing a probabilistic constraint, and enforcing a constraint on the conditional value at risk of the total number of unserved evacuees, respectively. We discuss the benefits and disadvantages of each, given certain carless population sizes and the variety of evacuation modes available. We also develop a heuristic approach to provide quick, dynamic and conservative solutions. We demonstrate the performance

of our approaches using five networks of varying sizes, and utilize real demographic data and hourly traffic count data to estimate the demand distribution.

For isolated, densely populated facilities, such as prisons and nursing homes, it is difficult to enact social distancing measures when catastrophic epidemiological outbreaks occur. In such facilities, strategic releases can enhance social distancing, yet have inherent costs, e.g., the potential for recidivism in crime for prisons, or the financial cost of incentives for residents to break contracts in nursing homes. In Chapter III, we examine how to structure these releases over time to de-densify isolated facilities under several competing objectives. We model the impact of strategic releases on infection transmission with a quadratic function that relates population size and daily interaction rate, which we call the de-densification function. Then, we formulate the decision problem as a multi-criteria MDP and develop dynamic solution methods that employ Monte Carlo simulations, k-means clustering, and Q learning with linear function approximation. We consider a 100-person facility experiencing an outbreak described by a Susceptible-Infectious-Recovered epidemiological model. Under this framework, we derive theoretical conditions for the de-densification function, to ensure it has an intuitive impact on infection transmission. We also test our dynamic solution methods under a number of parameter settings, and demonstrate that our cluster-based method outperforms a static benchmark by up to 13.3% under three different de-densification functions and two priority weights.

In Chapter IV, we model the flow of misinformation through a social media network from malicious user accounts to vulnerable populations with stochastic network interdiction. We minimize the expected number of vulnerable users reached by maximizing flow on a specific supergraph, and develop an adaptation for scalability based on community structure. We derive theoretical results about fake news mitigation on path networks and develop an approximate algorithm for the problem. We test our community-based model on a social network with 23,505 nodes, based on the IMDb

actors dataset.

CHAPTER I

Introduction

In our globalized world, with its growing population and ever-expanding interconnectedness, social crises are often large-scale and complex. Problem solving in this context requires interdisciplinary research, drawing from expertise in areas as far-reaching as sociology, economics, epidemiology, engineering, and mathematics. Operations research tools bridge this gap, optimizing discipline-driven responses to large-scale crises by capturing key features of complex problems with mathematical optimization models.

Marginalized communities are often disproportionately impacted by crises. Natural disasters particularly effect low-income communities, who are more likely to live in vulnerable geographical locations, as well as communities of color, senior citizens, and people with disabilities, who are less likely to have access to a personal vehicle for evacuation (*Zoraster et al.*, 2010). Epidemiological crises also heavily impact these groups, as seen in the COVID-19 pandemic (*Daoust*, 2020; *Tseng et al.*, 2020; *Van Dorn et al.*, 2020). Longer term cultural crises, like the rise of disinformation in social media (see, e.g., *Allcott and Gentzkow*, 2017), can also specifically affect marginalized groups, as hate speech more easily spreads (*Bilewicz and Soral*, 2020).

At the same time, if the needs of marginalized communities do not align with those in the general population, they may not be met by existing optimization models.

In evacuation planning, many models are adapted from traffic assignment models and aim to optimize vehicular flow by selecting evacuation routes (*Bayram*, 2016). These models can help the vast majority of a population evacuate quickly, but do not consider carless populations. In epidemiology and disease control applications, various optimization models developed during the COVID-19 pandemic weigh the benefits and disadvantages of social distancing and lockdown strategies (*Duque et al.*, 2020; *Rawson et al.*, 2020), and can identify approaches that balance competing factors while slowing the spread. On the other hand, these control models do not apply in situations where social distancing is not possible, including overcrowded facilities, like prisons and nursing homes. Higher risk populations are overrepresented in these facilities, and many have experienced outbreaks (*Brown et al.*, 2020; *Lemasters et al.*, 2020).

In this dissertation, we develop optimization models and solution methods for three community-centered intervention problems. In Chapter II, we consider volunteer recruitment and operations of an evacuation ridesharing program. Then, in Chapter III, we discuss strategic discharges for pandemic containment in overcrowded facilities. Finally, in Chapter IV, we explore mitigation of disinformation and hate speech dissemination in social media networks. The models and solution methods that we develop are specific to the crises and marginalized populations at hand, but many can be generalized. The algorithms we develop for facility de-densification during an epidemiological outbreak, for example, can be applied to a generic class of sequential decision making problems.

In all three projects, we utilize stochastic optimization models. These models are well-suited to intervention applications because they incorporate sequential decision making and future uncertainty. Use of these models requires careful consideration of risk measures, accurate quantifying of uncertainty, and operationalization of useful reward functions. The communities that the models consider are often large, which

requires development of scalable solution methods, as well.

In this chapter, we provide background on each of our three problems and outline the methods we utilize to describe and solve them in this dissertation.

1.1 Volunteer Recruitment and Operations for Evacuation Ridesharing

Resource planning for shared mobility based evacuation is situated in the field of emergency logistics (*Caunhye et al.*, 2012), and in particular, evacuation planning. Many evacuation planning models are adapted from traffic assignment models and optimize vehicular flow by selecting evacuation routes. Popular approaches include multi-objective evolutionary optimization (*Saadatseresht et al.*, 2009), simulation (*Gan et al.*, 2016), machine learning (*Tanaka et al.*, 2017), and heuristics (*Yusoff et al.*, 2008) for planning and facilitating evacuation operations. Some of these models incorporate uncertainty, although often in traffic parameters, rather than demand for services, which our models consider. *Lim et al.* (2019), for example, utilize a distributionally robust chance-constrained formulation to determine evacuation start time and traffic assignment under uncertain traffic demand. *Wang et al.* (2016) consider an evacuation routing problem under stochastic travel time and a variety of objective functions. For an overview of approaches for evacuation planning, we direct the readers to *Yusoff et al.* (2008).

Moreover, a wide spectrum of previous work has examined bus- or public transportation-based evacuation (see, e.g., *Bish*, 2011; *Kulshrestha et al.*, 2014; *Goerigk et al.*, 2015), two modes that can be effective for transporting large carless populations efficiently to safe zones and minimizing loss of life (*Litman*, 2006). At the same time, flexible, individualized modes of evacuation are also needed, to provide the personalized support some elderly carless populations and people with disabilities require to evac-

uate (*Renne et al.*, 2011). These populations may have difficulty getting to public transit or finding rides nearby. People with disabilities may require evacuation to specific facilities in a safe region with accommodations that serve their individual needs (*Lindell*, 2013) that may be different from the general population utilizing shuttles or buses for evacuation. Shared-mobility-based evacuation allows vulnerable carless populations to be picked up at home and taken to a particular destination during an evacuation (*Sadri et al.*, 2014). Additionally, in areas with less traffic congestion and smaller populations, where the benefits of transit-based evacuation are less pronounced, shared-mobility-based evacuation provides a flexible, low-cost option.

Whether shared-mobility-based evacuation is used as the main mode of evacuation for a particular population, e.g., people with disabilities, or included as a beneficial service to carless evacuees who can utilize transit if needed, it is crucial to plan ahead and ensure there are enough drivers to satisfy demand for services at the desired reliability level. Inadequate planning can lead to driver shortage when an emergency unfolds. Factors such as presence of family in the evacuation zone can reduce professional shared-mobility drivers' willingness to transport evacuees (*Li et al.*, 2018). For example, during Hurricane Katrina, 500 transit and school buses were available for evacuation, but city planners had difficulty finding drivers (*Litman*, 2006). In Chapter II, we introduce models to determine the number and location of volunteers to recruit ahead of time to ensure that evacuee demand for services is satisfied at a high reliability level.

Our formulation is closely related to and is a stochastic adaptation of the general pickup and delivery problem (GPDP) (*Savelsbergh and Sol*, 1995), which is a variant of the vehicle routing problem (VRP), aiming to minimize the distance traveled by a fleet of gasoline delivery trucks serving a set of gasoline stations. We refer interested readers to *Toth and Vigo* (2002) for a comprehensive review of VRP variants and models, as well as optimal/approximate algorithms or meta-heuristics for solving

them. While the VRP is often utilized in business applications, such as small package delivery (*Wong*, 2008), its variants are also used to solve problems of social benefit, including the Dial-a-Ride Problem (*Savelsbergh and Sol*, 1995; *Cordeau and Laporte*, 2007), Senior Transportation Problem (*Liu et al.*, 2018), and an integrated car-and-ride sharing problem, in which drivers without their own vehicles can rent cars from a company at a discounted price, in exchange for providing rides to other users without vehicles who are not able to drive (*Yu and Shen*, 2020).

1.2 De-Densification of Isolated Facilities for Pandemic Containment

Overcrowding is associated with the fast spread of a variety of infectious diseases, from tuberculosis to rheumatic heart disease to respiratory syncytial virus (RSV). In 2020, a study of 618 nursing homes in Ontario, Canada found that deaths from COVID-19 were more than doubled in facilities with high crowding conditions compared to those with low crowding conditions (*Brown et al.*, 2020). To combat overcrowding and enable social distancing, prisons throughout the United States released individuals with high health risk who were near the end of their sentences early in 2020 (*Cohen and Eisen*, 2022). Other types of densely populated facilities, like nursing homes, also may have benefited from temporary releases – with financial incentives for residents to stay with family or move to annex housing to reduce crowding conditions. While early prison release decisions tend to be ad hoc, determining those eligible for release and at greatest risk, Operations Research tools can help us better plan for future pandemics in our globalized world. This is key, as the world is experiencing an increase in the rate of pandemics, with Severe Acute Respiratory Syndrome (SARS), Influenza A H1N5, H1N1, Middle East Respiratory Syndrome (MERS), and Ebola, for example, all taking place over the decade from 2003-2014.

There are a few key features to this problem that should be captured by any decision-making tool. First, epidemiological outbreaks are dynamic phenomena with stochastic trajectories. Intuitively, the intervals at which facilities can release individuals, e.g., weekly or monthly, are discrete, each with immediate costs associated with infection and release. The decisions made at the beginning of each interval only need to consider the current state of the system, in terms of the outbreak and population demographics, without knowledge of previous states or decisions, in order to optimize some statistical measure of future cost. Eventually, an outbreak will subside. These features make the problem a good candidate for a discrete, finite-time Markov Decision Process (MDP).

When modeling de-densification for pandemic containment as an MDP, we need to operationalize the state space, action space, and reward structure. One common formulation of epidemiological outbreaks is the Susceptible-Infectious-Recovered (SIR) compartmental model, where each individual in a population is assumed to have S, I, or R status at any given time (*Kermack and McKendrick*, 1927). In our problem, as we balance epidemiological and release-related risk, we consider the state of the system to be a multidimensional vector with each entry the number of individuals with particular status, health risk, and release risk features. The actions are the number of susceptible individuals with particular health risk or release risk features released, and the reward is based on costs associated with release and costs associated with new infections for those who remain.

When we model the transition to a new epidemic state after a certain release action is taken, we use a generic underlying epidemiological model, but adaptively include the effect of the strategic release. Intuitively, reduced overcrowding enhances social distancing, which in turn, reduces daily interactions and finally, new infections. Thus, the epidemiological parameter directly affected by releases is the daily contact rate. For this reason, we model the release effect indirectly by modeling the contact rate as

a function of population size, in what we call the de-densification function. We will formally define the concept and examine its properties for a particular epidemiological model.

While generating the entire state space and transition probability matrix for a large-scale epidemiological MDP would be a Herculean task, constructing what *Kearns et al.* (2002) call a *generative model* of an epidemic is generally much more straightforward. A generative model is a black box that randomly samples reward and next state from any state-action pair, following the reward and transition probability distributions of the MDP. Any epidemiological simulator can be used as a generative model for the MDP developed later in Chapter III, with the initial SIR composition of the population given by subtracting the release action from the state. The simulator provides a new state at the end of the epoch, which includes information about the number of new infections, that can be used to calculate epidemiological cost.

This property of the strategic release problem leads us to the two solution approaches we consider in Chapter III. The first, based on a sparse sampling algorithm developed by *Kearns et al.* (2002), uses a lookahead tree to estimate the optimal action at a particular state. This method takes advantage of the fact that we have a generative model of the MDP available, sampling repeatedly from a single state, action pair in a way that purely model-free methods cannot, without requiring the full state-action transition probability matrix. This approach also does not require the same level of tuning of hyperparameters that an on-policy neural network would require. The second, Q learning with linear function approximation, estimates the Q function. While this offline method does not take advantage of the generative model in the same way, it has the benefit of producing a full policy, estimating the optimal action at any state. Like the sparse sampling method, it produces actions without having to construct the entire feasible state space or estimate state-action transition probabilities. This approach also can benefit from tailored basis functions

that take advantage of domain knowledge. In Chapter III, we demonstrate how the basis function can be formulated to mimic the generative model, incorporating the epidemiological dynamics in a different way.

1.3 Fake News Mitigation with Stochastic Network Interdiction

A plethora of research explores the problem of fake news detection in social media (*Shu et al.*, 2017), with detection methods based in style of content, characteristics of propagation, type of source, or fact checking (*Zhou and Zafarani*, 2020). *Zhou et al.* (2020), for example, represent news articles as vectors consisting of linguistic machine learning features, with the goal of classifying content as true or false based on writing style. *Zhang et al.* (2018) represent news articles, creators, and readers as a graph, aiming to learn the credibility of the content by using textual and connective features of the graph. Machine learning techniques can provide scalable approaches to fake news detection, but how should the spread of such information be mitigated, once detected? Because the mitigation problem is most interested in slowing the spread of information, rather than detection of the type of information, this problem can apply to both well-studied fake news and other types of harmful information, like hate speech or youth cyberbullying.

Two main threads in current fake news mitigation research are truth campaigning and influence minimization (*Saxena et al.*, 2022). In truth campaigning, social media networks aim to mitigate the influence of fake news by encouraging the spread of true news, countering misinformation (see, e.g., *Budak et al.*, 2011; *He et al.*, 2012; *Farajtabar et al.*, 2017). This type of research is more concerned with fake news in particular and has less applicability to hate speech mitigation.

Influence minimization, however, involves blocking a limited number of edges or

nodes in a network to minimize the number of users who receive misinformation, under an information diffusion model (*Nguyen et al.*, 2019), such as the linear threshold or independent cascade model (*Kempe et al.*, 2003). *Wang et al.* (2018) consider influence minimization in the context of vulnerable populations, aiming to protect specific target users from fake news with edge blocking, while incorporating budget constraints. *He et al.* (2011) consider both node and edge blocking, as they minimize a multi-criteria objective of infection cost and immunization cost. *Tanınmış et al.* (2020) consider a Stackelberg game, where the leader blocks a set of nodes to minimize influence spread, then the follower activates nodes to maximize it.

On the other hand, these papers in influence minimization assume that the effect of social network interventions – such as edge or node removal – is deterministic and fully effective. Interventions may not have the intended effect, however. Flagging news as "disputed" can have no effect or even positive effect on website traffic, for example (*Saxena et al.*, 2022). Our work in Chapter IV considers uncertainty in the effectiveness of interventions via a minimizing expected maximum flow version of stochastic network interdiction.

1.4 Structure of the Dissertation

The remainder of this dissertation is outlined as follows.

In Chapter II, we consider risk mitigation in evacuation planning for carless populations with specialized needs. In particular, we develop two-stage stochastic optimization models for volunteer recruitment for ridesharing-based evacuation that considers uncertainty in the spatial and temporal distribution of demand for services. These formulations minimize expected operational cost while mitigating the risk of unserved demand at three reliability levels, using an expected penalty cost, a probabilistic constraint, and a conditional value-at-risk constraint.

In Chapter III, we introduce a model to mitigate risk during a pandemic in iso-

lated, densely populated facilities, such as prisons and nursing homes, using strategic discharges to reduce overcrowding. We introduce a multi-criteria Markov Decision Process (MDP) that weighs epidemiological costs of overcrowding and costs associated with release. We develop solution approaches based on sparse lookahead trees, k-means clustering, and Q learning with linear function approximation.

In Chapter IV, we consider risk mitigation in social media networks, in particular, the spread of fake news or hate speech. We formulate the problem as an instance of stochastic network interdiction, constructing a graph that relates the expected number of vulnerable users reached in a social media network to expected maximum flow through the graph. We develop an adaptation for scalability that utilizes community structure present in many social networks. We derive theoretical results about fake news mitigation in path networks and demonstrate these results empirically. We implement our community-based adaptation, testing our solution methods on a social network based on the IMDb actors dataset.

CHAPTER II

A Shared Mobility Based Framework for Evacuation Planning and Operations under Forecast Uncertainty

2.1 Introductory Remarks

Planning for evacuations and ensuring efficient operations are key aspects of disaster relief and mitigation. In the United States, where transportation systems are designed around personal vehicle ownership, evacuation planning for carless populations can be particularly challenging (*Urbina and Wolshon*, 2003). Further difficulties in evacuation planning arise from the specialized assistance that carless populations, including people with disabilities and senior citizens, may rely on (*Renne et al.*, 2011). Because a variety of factors influence individual evacuation decisions, there is also uncertainty in the type of spatiotemporal evacuation patterns that will arise among these groups. It is important under this uncertainty to ensure that enough volunteer drivers are recruited from the right locations in an evacuation region to serve demand for ridesharing at a high reliability level. To address these issues, we propose a ridesharing program that can be incorporated into a short-notice evacuation plan, recruiting and matching volunteer drivers and their personal vehicles with evacuees without cars, and providing personalized assistance to those who need it. Specifically,

we develop models and solution methods for optimal resource planning for setting up the program in the presence of uncertain spatiotemporal demand. Our goal is to determine how many volunteer drivers to recruit before a disaster takes place, so that as many evacuees as possible can be served, when actual demand is uncertain but can be predicted with historical data. We also optimize the operations of such a program, which begin when demand for services is realized and volunteer drivers must be assigned to riders and routes before the disaster strikes. We assume that the probability distribution of spatiotemporal demand is fully known and obtainable from data, which is then used to sample finite scenarios to build the stochastic programming formulations.

We choose to make the above assumption about the full knowledge of the demand distribution, rather than utilizing a robust optimization approach under ambiguously known demand, due to the widespread usage of traffic cameras and frequency of short-notice evacuations in the United States. Practitioners can utilize population demographics, conduct surveys among carless populations, and analyze data from areas similar to their own to determine the appropriate probability distribution of spatiotemporal evacuation patterns in their community. The advantage of considering the probability distribution of spatiotemporal demand is that resource planning decisions can be made considering different levels of risk. In this chapter, we consider three different formulations to improve the proportion of evacuees served and discuss the benefits and disadvantages of each through extensive numerical studies.

We formulate this problem as a two-stage stochastic mixed-integer program (MIP), first with the objective of minimizing the first-stage volunteer recruitment cost and *expected* total distance traveled by all vehicles based on a set of demand scenarios sampled from a given distribution, while guaranteeing a specific proportion of evacuees are served in all scenarios. Then, we build an alternative version of the expectation-based stochastic program by using a chance-constrained formulation, to

determine how many volunteer vehicles to recruit in order to ensure that we can accommodate the same proportion of evacuation requests with sufficiently high probability. Lastly, we introduce an alternative risk-averse stochastic optimization model to the chance-constrained program, using a conditional value at risk (CVaR)-based formulation, which computes solutions more efficiently and yields solutions with performance guarantees. Table 2.1 provides an overview of all the formulations developed in this chapter, and descriptions about the notation in Table 2.1 can be found in Section 2.2. Following similar justifications in *Lu et al.* (2018); *Zhang et al.* (2021), we focus on two-stage stochastic optimization for planning resources in the first stage and evacuation operations in the second stage. We also build the two-stage chance-constrained formulation to ensure high probability of successful evacuation by coordinating planned resources (i.e., volunteer drivers hired) and emergency vehicle dispatch after realizing the evacuation demand. We discuss the merits of the three formulations and their respective computational results in Section 2.5.

2.1.1 Prior work on evacuation via ridesharing

The optimization of operations for shared-mobility-based evacuation has been studied with deterministic settings (*Naoum-Sawaya and Yu*, 2017; *Lu et al.*, 2020) and under uncertainty in travel time (*Li*, 2017; *Wang*, 2020). *Naoum-Sawaya and Yu* (2017) formulate the ridesharing problem as a mixed-integer program on a network. The spatial distribution of the demand is known, and the objective is to make routing decisions that maximize evacuees served in a fixed time. They also extend their model to minimize evacuation time and to include placement of extra vehicles at pickup nodes for evacuees to drive themselves. *Lu et al.* (2020) examine ridesharing-based evacuation as a particular instance of the Dial a Ride Problem with Transfers (DARPT). They match drivers and riders using a vehicle-space-time network and a multi-commodity network flow integer program. *Li* (2017) explores ridesharing-

Table 2.1: Overview and comparison of different risk measures and their corresponding formulations for evacuation planning.

	Expectation-based Formulation	Chance-constrained Formulation	CVaR-constrained Formulation
Description	Penalize any violation of evacuation threshold with high cost	Limit the probability that evacuation threshold is violated	Limit the expected value of worst-case differences between evacuation threshold and evacuees served
Best Suited For	Smaller carless populations, fewer evacuation modes	Larger carless populations, more evacuation modes	
Two-Stage Objective	$\min \mathbf{f}^T\mathbf{x} + \mathbb{E}_B[Q(\mathbf{x}, \mathbf{a}(\tilde{\omega}))]$	$\min\ \mathbf{f}^T\mathbf{x} + \mathbb{E}_B\left[\overline{Q}(\mathbf{x}, \mathbf{a}(\tilde{\omega}))\right]$	
Operational Phase Notation	$Q(\mathbf{x}, \mathbf{a}(\tilde{\omega}))$	$\overline{Q}(\mathbf{x}, \mathbf{a}(\tilde{\omega}))$	
Operational Phase Objective	Minimize distance traveled all vehicles plus penalty cost of unserved proportion $\mathcal{M}_e c_\omega$	Minimize distance traveled all vehicles	
Operational Evacuation Constraint	$\overline{q}_\omega \geq (\gamma - c_\omega)\overline{a}(\omega)$	None	
Two-Stage Constraints	$\mathbf{x} \in X \cap \{0,1\}^{\lvert S\rvert}$	$\mathbf{x} \in X \cap \{0,1\}^{\lvert S\rvert}$ $\mathbb{P}(\overline{q}_\omega \geq \gamma\overline{a}(\omega)) \geq 1 - \epsilon_0$	$\mathbf{x} \in X \cap \{0,1\}^{\lvert S\rvert}$ $\mathrm{CVaR}_{1-\epsilon_0}(\gamma\overline{a}(\omega) - \overline{q}_\omega) \leq 0$

based evacuation, using time windows to incorporate a temporal aspect to demand and specifying origins and destinations for both drivers and riders. The author models an integer program for matching drivers and riders and providing routing decisions. The work also uses a robust optimization formulation to incorporate uncertainty in travel time. Both of the deterministic integer program and robust formulation are solved using heuristics.

2.1.2 Contributions of the chapter

We study the effectiveness of three risk measures in the field of stochastic programming for shared-mobility-based evacuation planning and conduct computational tests in five networks of varying sizes, based on regions of Charleston County, South Carolina, an area that experienced a mandatory hurricane evacuation order in 2018. We estimate the geographical distribution of demand and volunteers available based

on granular demographic data and utilize hourly traffic counts over a two-week period that included the evacuation to estimate the temporal distribution of demand. In the planning phase, we decide the total number of volunteer vehicles to recruit, as well as their spatial distribution, in order to minimize volunteer recruitment costs and total expected distance traveled by all vehicles, while ensuring a high proportion of evacuation at different reliability levels. Incorporating stochastic demand requires optimizing over large sets of scenarios, leading to computational challenges in addition to solving the underlying integer program. Modeling realistic instance sizes increases the computational difficulty. We develop algorithms and methods to find high-quality solutions efficiently. Using results from our computational studies, we discuss the applicability of the different risk measures and their corresponding formulations depending on the size of carless population and amount of traffic congestion.

2.1.3 Structure of the chapter

The remainder of this chapter is organized as follows. In Section 2.2, we build the expectation-based, chance-constrained and CVaR-based stochastic programming formulations. In Section 2.3, we describe solution approaches for optimizing different models or deriving near-optimal solutions. In Section 2.4, we discuss the generation of test instances with parameter settings for our computational tests. In Section 2.5, we conduct computational studies and test instances of five regional networks, to demonstrate the efficacy of our approaches and also derive managerial insights of using ridesharing for evacuation. In Section 2.6, we summarize the research and discuss future research directions.

2.2 Mathematical Models

The ridesharing program that we propose has two phases. During the planning phase, volunteers with cars are recruited to provide rides in case of a disaster. Then,

when a short-notice evacuation order is given for a particular disaster, such as a hurricane, flood, or slowly-spreading wildfire, individuals who meet eligibility criteria can request rides at a specific location and time. Due to the variability of impact for different disasters, the volumes and geographical locations of demand for evacuation services are not known during the planning phase, and we are relying on an estimated distribution of the random demand. During the operational phase, once a particular instance of spatiotemporal demand is realized, decisions are made of where to route the volunteer drivers and vehicles, and how many evacuees to load at each location. Emergency vehicles are available to assist the volunteer vehicles and can be dispatched after we know the realized demand. Next, we define notation and assumptions, and summarize key notation in Table 2.2.

2.2.1 Notation, parameters and basic assumptions

Consider a geographical network, denoted $\mathcal{L} = (\mathcal{N}', \mathcal{A}')$, containing locations and connecting roadways in an evacuation region. Specifically, the set of locations is given by $\mathcal{N}' = \mathcal{N}'_P \cup \mathcal{O}' \cup \{h\}$, whose elements are denoted as follows. Evacuees can arrive at any pickup node $i \in \mathcal{N}'_P$ over a discrete set of time periods $t = 1, \ldots, T$. We have a set of emergency vehicles R and a set of volunteer vehicles S, whose union is defined to be V. Each emergency vehicle $r \in R$, has an origin node $o_r \in \mathcal{O}'$, and each volunteer vehicle $s \in S$ has an origin location $o_s \in \mathcal{O}'$. The location h represents a "safe zone." We assume that once a vehicle reaches the safe zone, it has left the evacuation region and is no longer available to pick up additional evacuees. In set $\mathcal{A}'$, there are arcs (o_v, j) for $v \in V = R \cup S$ and $j \in \mathcal{N}'_P$, arcs (i, j) for all $i, j \in \mathcal{N}'_P$ such that $i \neq j$, and arcs (i, h) for all $i \in \mathcal{N}'_P$. Each arc has weights t_{ij}, representing travel time from i to j, and d_{ij}, representing travel distance. If $t_{ij} = d_{ij} = 0$, the physical location of the two nodes is the same (e.g., a volunteer lives in a pickup zone). If $t_{ij} = d_{ij} = \infty$, nodes $i,\ j \in \mathcal{N}'$ are not connected in the evacuation network.

Table 2.2: A summary of notation and definitions.

Symbol	Definition
$\mathcal{G} = (\mathcal{N}, \mathcal{A})$	Spatiotemporal network made up of spatiotemporal nodes $\mathcal{N}$ and arcs $\mathcal{A}$
$\mathcal{L} = (\mathcal{N}'\mathcal{A}')$	Spatial network made up of spatial nodes $\mathcal{N}'$ and arcs $\mathcal{A}'$
$\mathcal{N}'_P$	Set of pickup locations in spatial network
$\mathcal{O}'$	Set of origin locations for vehicles
h	Safe-zone node in spatial network
R	Set of available emergency vehicles
S	Set of available volunteer vehicles
V	$R \cup S$
$t_{ij},\ d_{ij}$	Travel time and distance from location i to j in spatial network
$t = 1, \ldots, T$	Discrete time periods in which evacuees can be picked up
$i^t \in \mathcal{N}_P$	Spatiotemporal pickup node representing location $i \in \mathcal{N}'_P$ during time period t
$o_v^0 \in \mathcal{O}$	Spatiotemporal vehicle origin node for vehicle $v \in V$, with time period 0
h^{T+1}	Spatiotemporal safe zone sink node that represents the safe location h at time $T+1$
$\tilde{\omega}$	Random disaster variable
ω	A scenario; a particular instance of random disaster variable $\tilde{\omega}$
Ω_B	The support of random vector $\tilde{\omega}$
$\mathbf{a}(\tilde{\omega})$	Stochastic spatiotemporal demand that is a function of random disaster $\tilde{\omega}$
$\mathcal{Q}_v$	Maximum capacity of vehicle $v \in V$
x_s	Binary decision variable, whether to hire $s \in S$
$z_{r\omega}$	Binary decision variable, whether to hire $r \in R$ in scenario ω
$y_{vi^tj^{t'}\omega}$	Binary routing decision variable
$q_{vi^t\omega}$	Integer loading decision variable
$\alpha_s,\ \beta_r$	The cost to hire volunteer vehicle $s \in S$ or emergency vehicle $r \in R$
$\mathbf{f}^T\mathbf{x}$	Linear first stage recruitment cost of volunteer vehicles
X	Deterministic feasible region of volunteer vehicle hiring decisions
ζ	Period in $\{1, \ldots, 7\}$ of estimated Hurricane Florence evacuation
$\mu_\zeta^{\mathcal{N}'}$	Total expected demand for test instance comprising network $\mathcal{N}'$ and hourlong period ζ

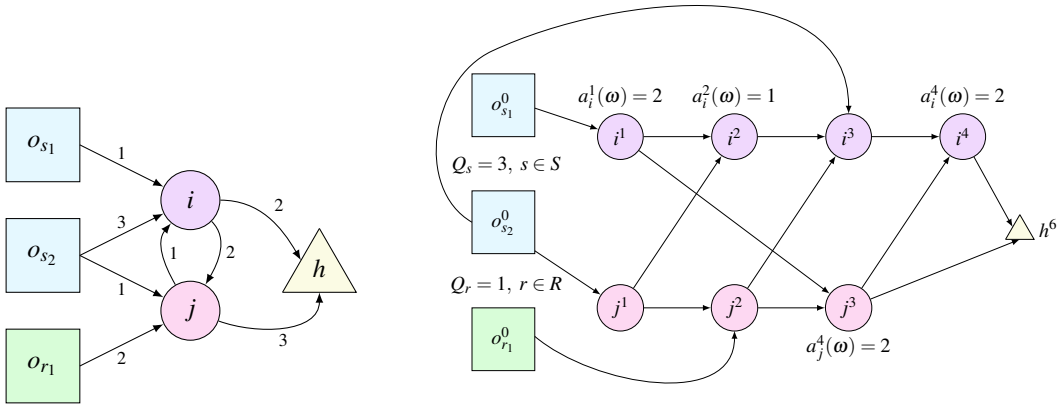
Figure 2.1a depicts one such geographical network with notation illustration, where we exclude arcs $(i, j) \in \mathcal{A}'$ having $t_{ij} = d_{ij} = \infty$.

To construct the corresponding spatiotemporal network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ based on a given $\mathcal{L}$, we create the set of spatiotemporal nodes $\mathcal{N} = \mathcal{N}_P \cup \mathcal{O} \cup \{h^{T+1}\}$, whose elements we explain as follows. Since evacuees can arrive throughout the planning horizon, we have pickup nodes $\mathcal{N}_P$ of the form i^t, for every $i \in \mathcal{N}'_P$ and $t = 1, \ldots, T$. We assume that all vehicles leave their origin node for a pickup node at time $t = 0$. Thus, we have a set of origin nodes $\mathcal{O} = \{o_v^0 : v \in V = R \cup S\}$. Also, all vehicles need to reach the safe zone at time $t = T + 1$, just after the last period that evacuees requesting service can arrive at a pickup location. Thus, the safe-zone node in $\mathcal{G}$ is h^{T+1}. (Note that we make these assumptions on the time of the origin nodes and safe zone to reduce the size of the network, but it is easy to convert the routing

and loading decisions described in later sections to an evacuation plan that includes start times $t = 1, \ldots, T$ for vehicles and times $t = 2, \ldots, T$ to reach the safe zone. Volunteers that wish to have later start times can be assigned these routes.) The set $\mathcal{A}$ of the spatiotemporal network contains idle arcs, connecting (i^t, i^{t+1}), along with travel arcs, $(i^t, j^{t'})$, that exist if the discretized travel time t_{ij} for $(i, j) \in \mathcal{A}'$ is equal to $t' - t$. We also have a set of dummy arcs $(h, o_v^0) \in \mathcal{A}$ for $v \in V$ with travel time and distance 0 to simplify flow balance constraints.

Spatiotemporal demand is a random vector $\mathbf{a}(\tilde{\omega}) = [a_j^t(\tilde{\omega}) : \ j^t \in \mathcal{N}_P]^\mathsf{T} \in \mathbb{Z}_+^{|\mathcal{N}_P|}$ that is a function of random disaster $\tilde{\omega}$, with entries $a_j^t(\tilde{\omega})$ equal to the number of arrivals at each spatiotemporal node $j^t \in \mathcal{N}_P$ requesting evacuation services. We denote the distribution of $\mathbf{a}(\tilde{\omega})$ as B and the support of $\tilde{\omega}$ as Ω_B. A realization ω of $\tilde{\omega}$ has fixed arrival vector $\mathbf{a}(\omega) \in \mathbb{Z}_+^{|\mathcal{N}_P|}$. Each vehicle $v \in V$ has a maximum capacity $\mathcal{Q}_v$, representing the number of evacuees requesting service that the driver can accommodate in a single ride.

Figure 2.1 illustrates the correspondence between a geographical network and its spatialtemporal transformation, where spatial network $\mathcal{L} = (\mathcal{N}', \mathcal{A}')$ pictured in Figure 2.1a has $\mathcal{N}_P' = \{i, j\}$, $S = \{s_1, s_2\}$, $R = \{r_1\}$, and arcs labeled by travel time t_e for $e \in \mathcal{A}'$. (We exclude all the arcs e with $t_e = \infty$.) The spatiotemporal network constructed from this spatial network, assuming $T = 5$, is pictured in Figure 2.1b with description of the capacities of the vehicles, and one realization of demand $\mathbf{a}(\omega)$. One possible evacuation plan for this example is as follows. Volunteer vehicle s_1 travels to i^1 and picks up two people, then travels to j^3 and picks up one person. Another volunteer vehicle s_2 travels to i^3 and picks up one person who has been waiting since $t = 2$. The vehicle waits, traveling along the idle arc to i^4, and picks up two more people at i^4. Emergency vehicle r_1 drives to j^2 and waits until $t = 3$ and picks up one person at j^3. All three vehicles travel to the safe-zone node h^6, by time $T + 1 = 6$, after picking up their last passengers.



(a) (b)

Figure 2.1: An example of (a) geographical network $\mathcal{L}$ with 6 locations, labeled with travel time t_e, $\forall e \in \mathcal{A}'$ and (b) its spatiotemporal network $\mathcal{G}$ with $T = 5$, demand realization $\mathbf{a}(\omega)$ and vehicle capacities $\mathcal{Q}_v$ for $v \in V$.

2.2.2 Expectation-based formulation

We define planning-phase binary decision variables, $\mathbf{x} = [x_s,\ s \in S]^\mathsf{T} \in \{0,1\}^{|S|}$, where $x_s = 1$ if we hire volunteer vehicle $s \in S$, and $x_s = 0$ otherwise. These decisions take place before realizing the random spatiotemporal demand $\mathbf{a}(\tilde{\omega})$, with knowledge of its distribution B. They yield a deterministic linear cost, $\mathbf{f}^T\mathbf{x}$, where cost vector $\mathbf{f}$ has the same dimension as decision vector $\mathbf{x}$. We keep the cost function generic, to represent the planner's preferences. For instance, this cost can involve incentives to recruit volunteers, with higher incentives given to volunteers in areas known to have high demand or low supply. However, in our computational studies, we are most interested in the number and distribution of volunteers required to satisfy demand for services at a specific reliability level. For this reason, we assume a flat cost for volunteers and use $\mathbf{f}$ as a weight vector to balance volunteer recruiting cost and expected travel distance in the overall objective described later.

After a disaster takes place and realization $\mathbf{a}(\omega)$ is known, the operational phase occurs. At this time, operational decisions are made, with respect to both $\mathbf{x}$ and $\mathbf{a}(\omega)$,

with the goal of serving at least $\gamma \in (0, 1]$ proportion of evacuees, while minimizing second-stage cost – total distance traveled by all vehicles. Note that $\gamma = 1$ corresponds to full evacuation. In some settings, if multiple modes of transportation are available for carless populations, a smaller parameter may be appropriate.

In the operational phase, for each scenario $\omega \in \Omega_B$, we define binary variables $\mathbf{z}_\omega = [z_{r\omega},\ r \in R]^\mathsf{T} \in \{0,1\}^{|R|}$ to indicate hiring decisions for emergency vehicles $r \in R$, where $z_{r\omega} = 1$ if emergency vehicle r is hired in scenario ω and $z_{r\omega} = 0$ otherwise. We define binary variables $\mathbf{y}_\omega = [y_{vi^tj^{t'}\omega},\ v \in V,\ (i^t, j^{t'}) \in \mathcal{A}]^\mathsf{T} \in \{0,1\}^{|V|\times|\mathcal{A}|}$ to represent routing decisions, where $y_{vi^tj^{t'}\omega} = 1$ if vehicle $v \in V$ travels the arc $(i^t, j^{t'}) \in \mathcal{A}$ in scenario ω and 0 otherwise, along with integer variables $\mathbf{q}_\omega = [q_{vi^t\omega},\ v \in V,\ i^t \in \mathcal{N}_P]^\mathsf{T} \in \mathbb{Z}^{|V|\times|\mathcal{N}_P|}$ that represent loading decisions, where $q_{vi^t\omega}$ is the number of passengers picked up by vehicle $v \in V$ at node $i^t \in \mathcal{N}_P$ in scenario ω. For each scenario $\omega \in \Omega$, we define an integer variable $\overline{q}_\omega \in \mathbb{Z}$ to represent the total number of evacuees picked up in the scenario, at all spatiotemporal nodes. In order to ensure that the second-stage problem is feasible for any first-stage solution $\mathbf{x}$, we define continuous variable $c_\omega \in [0, \gamma]$ that represents the proportion of evacuees not served, less $1-\gamma$, and associate with it a high penalty cost $\mathcal{M}_e$. To simplify notation, we denote the total demand for scenario ω as $\overline{a}(\omega) = \sum_{t=0}^{T} \sum_{j\in\mathcal{N}_P'} a_j^t(\omega)$. Thus, for each scenario $\omega \in \Omega_B$, given a planning solution $\mathbf{x}$ and demand realization $\mathbf{a}(\omega)$, the corresponding operational phase optimization problem can be modeled as follows:

$$Q(\mathbf{x}, \mathbf{a}(\omega)) = \min_{\mathbf{y}_\omega, \mathbf{z}_\omega, \mathbf{q}_\omega} \sum_{v\in V} \sum_{(i^t, j^{t'})\in A} y_{vi^tj^{t'}\omega} d_{ij} + \mathcal{M}_e c_\omega \tag{2.1a}$$

$$\text{s.t. } \overline{q}_\omega \geq (\gamma - c_\omega)\overline{a}(\omega) \tag{2.1b}$$

$$0 \leq c_\omega \leq \gamma \tag{2.1c}$$

$$\overline{q}_\omega = \sum_{v\in V} \sum_{j^t\in\mathcal{N}_P} q_{vj^t\omega} \tag{2.1d}$$

$$\sum_{v\in V} q_{vj^t\omega} \leq \sum_{t=0}^{t'} a_j^t(\omega) - \sum_{v\in V} \sum_{t=0}^{t'-1} q_{vj^t\omega},\ j^{t'} \in \mathcal{N}_P \tag{2.1e}$$

$$q_{vj^{t'}\omega} \leq \mathcal{Q}_v \sum_{i^t:(i^t,j^{t'})\in\mathcal{A}} y_{vi^tj^{t'}\omega},\ j^{t'} \in \mathcal{N}_P, v \in V \tag{2.1f}$$

$$\sum_{j^{t'}\in\mathcal{N}_P} q_{vj^t\omega} \leq \mathcal{Q}_v,\ v \in V \tag{2.1g}$$

$$\sum_{j^t:(o_s^0,j^t)\in\mathcal{A}} y_{so_s^0j^t\omega} = x_s,\ s \in S \tag{2.1h}$$

$$\sum_{j^t:(o_r^0,j^t)\in\mathcal{A}} y_{ro_r^0j^t\omega} = z_{r\omega},\ r \in R \tag{2.1i}$$

$$\sum_{i^t:(i^t,j^{t'})\in\mathcal{A}} y_{vi^tj^{t'}\omega} - \sum_{i^t:(j^{t'},i^t)\in\mathcal{A}} y_{vj^{t'}i^t\omega} = 0,\ j^{t'} \in \mathcal{N}, v \in V \tag{2.1j}$$

$$\sum_{s\in S} \alpha_s x_s + \sum_{r\in R} \beta_r z_{r\omega} \leq C \tag{2.1k}$$

$$\mathbf{z}_\omega \in \{0,1\}^{|R|},\ \ \mathbf{y}_\omega \in \{0,1\}^{|V|\times|\mathcal{A}|},\ \ \mathbf{q}_\omega \in \mathbb{Z}_+^{|V|\times|\mathcal{N}_P|}. \tag{2.1l}$$

In the objective function (2.1a), we minimize the total distance traveled by all vehicles plus the penalty associated with violating the evacuation threshold. Given appropriate value of $\mathcal{M}_e$, constraint (2.1b) ensures that $\gamma \in (0,1]$ proportion of evacuees are served whenever feasible. To see this, note that the left hand side of the constraint is the number of evacuees served. The right hand side is $(\gamma - c_\omega)$ proportion of total demand in scenario ω. Constraint (2.1c) ensures that c_ω is between 0 and desired proportion of evacuees served γ. Because of the high cost associated with variable c_ω in the objective, whenever possible, $c_\omega = 0$ and the right hand side will be the evacuation threshold (i.e., γ proportion of total demand).

Constraint (2.1d) ensures that variable $\overline{q}_\omega$ represents the total number of evacuees served. Constraints (2.1e) ensure that the number of passengers picked up at node j^t by any vehicle is less than or equal to those that have arrived by time t and have not yet been picked up in earlier periods. Constraints (2.1f) ensure that vehicle v only picks up passengers at node $j^{t'}$ if it drives to node $j^{t'}$. Constraints (2.1g) ensure that no vehicle picks up more passengers than its capacity, $\mathcal{Q}_v$. Constraints (2.1h)–(2.1i) ensure that a vehicle v leaves its origin node o_v^0 if and only if it is hired in that scenario.

Constraints (2.1j) are flow balance constraints to ensure that when a vehicle v enters node $j^{t'}$, it also exits that node. Constraint (2.1k) is a budget constraint, ensuring that the total cost, calculated based on unit cost α_s of hiring volunteer vehicle $s \in S$ and unit cost β_r of hiring emergency vehicle $r \in R$, is less than or equal to a given budget parameter C.

The overall objective is to minimize the weighted sum of first stage recruitment cost and the expected distance traveled by all vehicles, while ensuring γ proportion of evacuees are served in all scenarios whenever possible with respect to the distribution B. We define set X as a deterministic feasible region that encompasses any constraints on the volunteer vehicles hired. For example, X can contain upper-bound capacities on the number of volunteers we can hire at each location. One can also include the correlations between the number of volunteers across different locations in the set X, if such correlations can be obtained a priori and approximated through linear constraints. The function $\mathbf{f}^T\mathbf{x}$ is a weighted linear cost function associated with first-stage recruitment, and $\mathbb{E}_B[\cdot]$ denotes the expectation of $\cdot$ under distribution B of the uncertain parameter. Then the overall problem is given by

$$u^* = \min_{\mathbf{x}} \mathbf{f}^T\mathbf{x} + \mathbb{E}_B[Q(\mathbf{x}, \mathbf{a}(\tilde{\omega}))] \qquad \text{(Exp)}$$
$$\text{s.t. } \mathbf{x} \in X \cap \{0,1\}^{|S|}.$$

Solving (Exp) exactly is computationally difficult, even if the support of the distribution is discrete and finite. Further difficulty arises if the support is continuous, for which (Exp) involves multidimensional integration. For this reason, we utilize the Sample Average Approximation (SAA) approach (*Kleywegt et al.*, 2002). In SAA, a Monte Carlo sample Ω of random variable $\tilde{\omega}$ of size N' is taken. (Note that Ω contains samples of the uncertain parameter, as opposed to Ω_B, which denotes the support of the probability distribution.) Because each $\omega \in \Omega$ is independently sampled, we

have $\frac{1}{N'}\sum_{n=1}^{N'} Q(\mathbf{x}, \mathbf{a}(\omega^n)) \to \mathbb{E}_B[Q(\mathbf{x}, \mathbf{a}(\tilde{\omega}))]$ almost surely as $N' \to \infty$, and the SAA reformulation is

$$\min_{\mathbf{x}} \mathbf{f}^T\mathbf{x} + \frac{1}{N'}\sum_{n=1}^{N'} Q(\mathbf{x}, \mathbf{a}(\omega^n)) \qquad \text{(Exp-SAA)}$$
$$\text{s.t. } \mathbf{x} \in X \cap \{0,1\}^{|S|}.$$

Kleywegt et al. (2002) prove a bound for the sample size N' required to ensure that the optimal objective value of (Exp-SAA) is within ϵ of the true optimum u^* with high probability. The bound is conservative, though, requiring a very large sample size N', which can be computationally prohibitive. On the other hand, solving $|\Omega| = N'$ subproblems $Q(\mathbf{x}, \mathbf{a}(\omega^n)),\ n = 1 \ldots, N'$ is often much less computationally expensive than minimizing $\frac{1}{N'}\sum_{n=1}^{N'} Q(\mathbf{x}, \mathbf{a}(\omega^n))$ over the deterministic feasible region. For this reason, we take M Monte Carlo samples of smaller size N, where each sample set $\Omega_m = \{\omega_m^1, \ldots, \omega_m^N\}$, to find an optimal solution $\hat{\mathbf{x}}_m$ to (Exp-SAA) with an objective value $\hat{u}_m$, for $m = 1, \ldots, M$. Let $\overline{u} = \frac{1}{M}\sum_{m=1}^{M} \hat{u}_m$. *Mak et al.* (1999) show that $\mathbb{E}[\overline{u}] \leq u^*$, the optimal objective value of (Exp), and we can find an estimated lower bound of the true optimum. These M independent optimization processes are referred to as the in-sample tests.

Because any candidate solution $\hat{\mathbf{x}}_m,\ m = 1, \ldots, M$ is feasible for any realization of $\tilde{\omega}$ (by letting $\mathbf{q}_\omega = \mathbf{z}_\omega = \mathbf{0}$ and $c_\omega = \gamma$ in Model (Exp) for all scenarios $\omega \in \Omega_B$), we have an upper bound $\mathbb{E}_B[Q(\hat{\mathbf{x}}_m, \mathbf{a}(\tilde{\omega}))] \geq u^*,\ \forall m = 1, \ldots, M$. Using a sample $\Omega_0 = \{\omega_0^1, \ldots, \omega_0^{N'}\}$ of size $N' > N$ that is independent of the samples $\{\Omega_m\}_{m=1}^M$, we can estimate the expectation with $\frac{1}{N'}\sum_{n=1}^{N'} Q(\hat{\mathbf{x}}_m, \omega_0^n)$. Because this value is an unbiased estimator of $\mathbb{E}_B[Q(\hat{\mathbf{x}}_m, \mathbf{a}(\tilde{\omega}))]$, we have a statistical upper bound $\frac{1}{N'}\sum_{n=1}^{N'} Q(\hat{\mathbf{x}}_m, \mathbf{a}(\omega_0^n)) \geq u^*$, for each $m = 1, \ldots, M$. As the result holds for all $\hat{\mathbf{x}}_m,\ m = 1, \ldots, M$, we can pick the candidate solution with the lowest objective value as the best upper bound. To ensure we have an unbiased estimator, we perform

the test again with the chosen candidate solution and a second independent sample $\overline{\Omega}_0 = \{\overline{\Omega}_0^1, \ldots, \overline{\Omega}_0^{N'}\}$, also of size N'. Thus, our final estimate of the upper bound expectation is $\frac{1}{N'} \sum_{n=1}^{N'} Q(\hat{\mathbf{x}}_m, \overline{\omega}_0^n)$. The processes of computing the objective upper bound are called the out-of-sample tests.

Methods for improving the SAA. Several enhancement strategies from the existing literature can be used for facilitating the SAA process. We divide them into the following two streams. The first stream of studies concentrates on improving the isolated solution process for urging quick convergence to optimal or sub-optimal solutions of Monte Carlo samples (i.e., the M in-sample tests, each with size N) and the second independent out-of-sample test with size N'. For example, *Khalil et al.* (2016) introduce a novel data-driven framework for variable selection to solve MIPs via a branch-and-bound algorithm; *Kruber et al.* (2017) develop a supervised learning approach to distinguish a stronger reformulation of a given MIP, to improve the speed of MIP solvers; *Jia and Shen* (2021) develop a learning-enhanced Benders decomposition algorithm specifically for solving two-stage stochastic programs; and *Deng et al.* (2021) develop a bilevel program to optimally groups scenarios to tighten the quantile bounds for chance-constrained programs. (We will discuss the SAA-based reformulation for the chance-constrained problem in Section 2.2.3). The second stream of the literature studies scenario reduction approaches, which can be interpreted as decreasing the values of N and N' to speed up the solution process while preserving the solution quality. For example, *Rujeerapaiboon et al.* (2018) establish performance guarantees for Wasserstein-distance-based scenario reduction; similarly, *Keutchayan et al.* (2021) consider scenario reduction by the cost difference of scenarios; *Bertsimas and Mundru* (2022) develop a novel optimization-based approach for scenario selection by minimizing "problem-dependent divergence" between the nominal distribution and selected scenarios; and *Arpón et al.* (2018) propose an algorithm that finds the relevant scenarios from among the original empirical scenarios specifically

considering a CVaR objective. (We discuss the SAA reformulation for the CVaR model in Section 2.2.4.)

While it is possible to perform iterative tests to narrow the optimality gap between statistical lower and upper bounds on the optimal objective value u^*, in this chapter, we are more interested in finding quality, reliable evacuation solutions using stochastic optimization. For this reason, we emphasize out-of-sample performance metrics, such as probability of full evacuation, number of volunteers recruited, and mean total distance traveled, to evaluate solutions, rather than the optimality gap results in our numerical studies. We will consider further improvements of the SAA method in the future research.

2.2.3 Chance-constrained program

A key feature of (Exp-SAA) in Section 2.2.2 is its heavy preference for candidate solutions that serve γ proportion of evacuees in every out-of-sample scenario $\omega \in \Omega_0$. When γ is high (e.g., close to 1), it can skew the solutions towards serving rare patterns of spatiotemporal demand, such as uncharacteristically large total demand or spatial dispersion of demand. In areas with larger carless populations, where bus or transit-based evacuation methods are crucial, ridesharing acts as an additional, beneficial service rather than the main mode of evacuation. For these settings, where a less conservative solution may be appropriate, we introduce a chance-constrained version of the stochastic program (*Charnes and Cooper*, 1959), in which probabilistic constraints are used to ensure that the quality-of-service threshold is met with high probability.

In this model, rather than serving γ proportion of evacuees in all scenarios whenever possible, our goal is to satisfy the following probabilistic constraint:

$$\mathbb{P}_B\left(\overline{q}_{\tilde{\omega}} \geq \gamma \overline{a}(\tilde{\omega})\right) \geq 1 - \epsilon_0, \tag{2.2}$$

which ensures with probability at least $1-\epsilon_0$ that at least γ proportion of evacuees requesting services are successfully evacuated, with respect to the distribution B of $\mathbf{a}(\tilde{\omega})$. The evacuation threshold constraint relates to the entire support of the distribution, restricting the probability of violations. For this reason, the evacuation threshold constraint is a first-stage constraint, as opposed to a scenario-specific operational phase constraint in (Exp). We also do not utilize a slack variable or associated penalty cost for unserved demand in this formulation, instead mitigating risk with the probabilistic constraint. Thus, the operational phase problem for each realized scenario ω is adapted as follows.

$$\overline{Q}(\mathbf{x}, \mathbf{a}(\omega)) = \min \sum_{v \in V} \sum_{(i^t, j^{t'}) \in A} y_{vi^t j^{t'} \omega} d_{ij} \tag{2.3}$$

$$\text{s.t. (2.1d)–(2.1l).}$$

Then, the chance-constrained stochastic program can be written as follows.

$$\min \mathbf{f}^T \mathbf{x} + \mathbb{E}_B \left[\overline{Q}(\mathbf{x}, \mathbf{a}(\tilde{\omega})) \right] \tag{CC}$$

$$\text{s.t. } \mathbf{x} \in X \cap \{0,1\}^{|S|}, (2.2).$$

Note that the objective of the two-stage chance-constrained model remains to minimize recruitment costs plus expected distance traveled. Without the penalty cost, however, the distance traveled in scenarios that violate the quality of service threshold is minimized to the distance traveled by hired volunteer vehicles leaving the evacuation region.

Following the SAA approach described in *Luedtke and Ahmed* (2008), we approximate the probabilistic constraint (2.2) using the empirical distribution of the Monte Carlo sample Ω, which has finite cardinality. We model this empirical probabilistic constraint using binary decision variables $\mathbf{k} = [k_\omega : \omega \in \Omega]^\mathsf{T} \in \{0,1\}^{|\Omega|}$, where $k_\omega = 1$ if $\overline{q}_\omega \geq \gamma \overline{a}(\omega)$ is violated for scenario ω and $k_\omega = 0$ otherwise. Then, we have the

following constraints:

$$\overline{q}_\omega \geq \gamma \overline{a}(\omega)(1 - k_\omega), \quad \forall \omega \in \Omega, \tag{2.4a}$$

where the evacuation threshold only must be satisfied if $k_\omega = 1$, and

$$\sum_{\omega \in \Omega} k_\omega \leq \lfloor \epsilon |\Omega| \rfloor, \tag{2.4b}$$

which ensures that the probabilistic constraint is satisfied for a constant $\epsilon \leq \epsilon_0$.

Thus, we can formulate the chance-constrained model as a MIP as follows:

$$\min_{\mathbf{x}} \ \mathbf{f}^T\mathbf{x} + \frac{1}{N}\sum_{n=1}^{N}\sum_{v \in V}\sum_{(i^t, j^{t'}) \in A} y_{vi^t j^{t'} \omega^n} d_{ij} \tag{CC-SAA}$$

$$\text{s.t.} \ \ (2.1\text{d})\text{–}(2.1\text{l}), \forall \omega \in \Omega, \ (2.4\text{a}), \ (2.4\text{b}).$$

Luedtke and Ahmed (2008) show that feasible solutions to (CC-SAA) satisfy the probabilistic constraint (2.2) with probability that goes to 1 as $|\Omega| = N \to \infty$. Again, because the sample size N required to ensure with high confidence that the solutions to (CC-SAA) satisfy (2.2) is prohibitively high, we follow a similar method to that described in Section 2.2.2, of drawing M Monte Carlo samples $\Omega_1, \ldots, \Omega_M$ of size N. We then solve (CC-SAA) with each scenario set Ω_m to obtain its optimal solution $\hat{\mathbf{x}}_m$, for $m = 1, \ldots, M$. Then, with a larger, independent Monte Carlo sample of size $N' > N$, we check if the candidate solutions $\hat{\mathbf{x}}_m$ can satisfy the quality of service threshold for more than $(1 - \epsilon_0)$ proportion of scenarios $\{\omega^n\}_{n=1}^{N'}$, an approximation that is stronger, on expectation, than the original approximations.

Specifically, we find the optimal value, $\overline{Q}(\mathbf{x}_m, \mathbf{a}(\omega^n))|_{\overline{q}_\omega \geq \gamma \overline{a}_\omega}$, $n = 1, \ldots, N'$, for the second stage problem when we force the quality of service threshold to be satisfied for each scenario. If these values are finite (i.e., the second stage problem with evacuation

threshold constraint is feasible) for $\lceil(1-\epsilon_0)|\Omega_0|\rceil$ scenarios, then the candidate solution is feasible to the approximation of the probabilistic constraint.

To determine which feasible candidate solution is best, we calculate the objective of (CC-SAA) using out-of-sample scenarios. We choose the best $\lceil(1-\epsilon_0)|\Omega_0|\rceil$ finite values $\overline{Q}(\mathbf{x}, \mathbf{a}(\omega^n))|_{\bar{q}_\omega \geq \gamma \bar{a}_\omega}$, take any remaining scenarios and find the optimal value $\overline{Q}(\mathbf{x}_m, \mathbf{a}(\omega^n))$ to the second stage problem when we require no evacuees to be served. Because there are hired volunteer vehicles that must leave the evacuation zone, these values are positive but much smaller. We use these values to calculate the final out-of-sample objective value. We take the best solution to be the "best" feasible solution, which is the candidate solution having the lowest cost that satisfies the approximation of the probabilistic constraint as in *Luedtke and Ahmed* (2008). In our computational tests, to directly compare the performance of the chosen candidate solution with that of (Exp-SAA), we perform another out-of-sample test using an independent sample as described in Section 2.2.2.

Remark II.1. We choose to have a different procedure to select the best candidate and then to evaluate the performance of the candidate for the following reasons. The chance-constrained method allows us to pick the planning solution that satisfies the quality of service constraint with high probability, allowing lower quality of service in rare scenarios. Thus, when selecting which candidate is best in this method, we allow the candidate to perform as poorly as possible in the worst-case scenarios. Here, we are choosing the best solution with respect to the objective of the chance-constrained model. Since the objective differs from the objective of the expectation-based model, the procedure differs from that used in Section 2.2.2. On the other hand, when implementing the planning solution, when a disaster strikes and demand is realized, as many people as possible are desired to be served. Thus, we evaluate the final performance of the chosen candidate solution by using the expected-penalty cost formulation. This also allows a more direct comparison with (Exp-SAA). We

demonstrate the computational efficacy of the SAA-based chance-constrained model later in Section 2.5.2.

2.2.4 CVaR approximation of chance-constrained program

We approximate the chance constraint (2.2) in Section 2.2.3 via conditional value-at-risk (CVaR). Letting Y be a general random variable, we consider the definitions of two risk measures

$$\mathrm{VaR}_{1-\epsilon_0}(Y) = \inf\{\tau | \mathbb{P}_B(Y \leq \tau) \geq 1 - \epsilon_0\},$$

and

$$\mathrm{CVaR}_{1-\epsilon_0}(Y) = \mathbb{E}_{\mathbb{P}_{1-\epsilon_0}}[Y] \text{ with } \mathbb{P}_{1-\epsilon_0} = \begin{cases} 0, & \text{if } y < \mathrm{VaR}_{1-\epsilon_0}(Y) \\ \dfrac{\mathbb{P}(y) - (1-\epsilon_0)}{\epsilon_0} & \text{if } y \geq \mathrm{VaR}_{1-\epsilon_0}(Y) \end{cases}$$

From the definitions we can easily observe that VaR can be bounded from above by CVaR. Further, CVaR can be equivalently expressed by the following program (see, e.g., *Rockafellar et al.* (2000); *Rockafellar and Uryasev* (2002)):

$$\mathrm{CVaR}_{1-\epsilon_0}(Y) = \min_y \left\{ y + \frac{1}{\epsilon_0}\mathbb{E}[Y - y]^+ \right\}. \tag{2.5}$$

Therefore, the probabilistic constraint (2.2) can be guaranteed by requiring

$$\mathrm{CVaR}_{1-\epsilon_0}\left(\gamma \overline{a}(\tilde{\omega}) - \overline{q}_{\tilde{\omega}}\right) \leq 0. \tag{CVaR}$$

Consider a finite set of Monte Carlo samples Ω. We define a nonnegative continuous decision variable κ_ω for scenario $\omega \in \Omega$, and an auxiliary continuous variable τ. (Note that the continuous variables κ_ω are distinct from the binary variables k_ω, defined for $\omega \in \Omega$ in (CC-SAA).) Based on (2.5), we obtain the following SAA reformulation of

(CVaR):

$$\tau + \frac{1}{\epsilon|\Omega|}\sum_{\omega\in\Omega}\kappa_\omega \leq 0 \tag{2.6a}$$

$$\gamma\overline{a}(\omega) - \overline{q}_\omega - \kappa_\omega \leq \tau, \quad \forall\omega \in \Omega, \tag{2.6b}$$

$$\kappa_\omega \geq 0, \quad \forall\omega \in \Omega, \tag{2.6c}$$

ensuring that Constraint (CVaR) over the sample set Ω is satisfied for a constant $\epsilon \leq \epsilon_0$.

Replacing (2.4a) and (2.4b) with (2.6) in (CC-SAA), we reformulate

$$\min_{\mathbf{x},\tau,\kappa} \ \mathbf{f}^T\mathbf{x} + \frac{1}{N}\sum_{n=1}^{N}\sum_{v\in V}\sum_{(i^t,j^{t'})\in A} y_{vi^tj^{t'}\omega^n}d_{ij} \tag{CC-CVaR-SAA}$$

$$\text{s.t.} \ \ (2.1\text{d})\text{–}(2.1\text{l}), \forall\omega \in \Omega, \ (2.6).$$

Similar to (CC-SAA), (CC-CVaR-SAA) has a CVaR constraint for the number of evacuees served by the system and the feasible solutions satisfy (CVaR) with probability 1 as the sample size approaching infinity. We again apply the SAA approach to (CC-CVaR-SAA), with in-sample tests of size N and out-of-sample simulation tests of larger size N'. Because the CVaR formulation is used as an approximation of the chance-constrained formulation, for all candidate out-of-sample tests, we utilize the same procedures as those in Section 2.2.3, including checking whether the probabilistic constraint is satisfied and calculating an approximation of the objective of (CC-SAA).

Note that (CC-CVaR-SAA) only requires the use of new continuous, rather than binary variables. Because the function in (CVaR) includes integer variables, it is not convex, and the final constraints in (CC-CVaR-SAA) are nonconvex. Meanwhile, the addition of continuous variables rather than binary variables to the model can

improve computational performance. Therefore, the CVaR reformulation provides a more conservative solution which yields an upper bound of the optimal objective value of (CC-SAA), while improving the computational tractability from the respect of modeling formulation. (Note that CVaR itself is not a conservative risk measure; rather, for the same risk parameter ϵ and sample Ω, (CC-CVaR-SAA) provides a conservative solution to (CC-SAA). Though we use it as an approximation to the chance-constrained model in our numerical studies, CVaR is a practical risk measure with its own benefits (see relevant discussions in *Rockafellar et al.* (2000); *Rockafellar and Uryasev* (2002)).

2.3 Solution Algorithm

We further introduce a heuristic-based algorithm to find risk-averse candidate solutions when desired proportion of evacuation $\gamma = 1$. This method can be used as an alternative to (Exp-SAA), our most risk-averse formulation, for large problem sizes. We demonstrate the computational performance of directly solving (Exp-SAA) using off-the-shelf optimization solvers in Section 2.5, showing that reliable solutions can be found in a few hours for test instances with up to 16 locations and 4 time periods, with $|\Omega_m| = 50$, for each in-sample $m = 1, \ldots, M$, with a relaxation of (Exp-SAA) and scenario-decomposition procedure to find initial feasible solutions. On the other hand, in this problem, volunteer drivers with their own vehicles are low-cost and fairly easy to recruit, which implies quick, conservative solutions could also provide useful information to decision makers. For larger problem sizes that cannot be accommodated by an integer program formulation, heuristic-based approaches can be especially helpful for improving evacuation efficiency and also planning under uncertainty.

We develop an algorithm that constructs a single deterministic worst-case scenario from all scenarios in a Monte Carlo sample Ω_m to find a feasible solution $\hat{\mathbf{x}}_m$ to

(Exp-SAA), $m = 1, \ldots, M$. Later, we measure the performance of this approach using second-stage function $Q(\hat{\mathbf{x}}_m, \mathbf{a}(\omega))$ that includes penalty cost for unserved demand, described in Section 2.2.2, performing out-of-sample tests for each scenario $\omega \in \Omega_0$, an independent Monte Carlo sample of size $N' > N = |\Omega_m|$. In order to compare out-of-sample performance of the heuristic with (Exp-SAA), we choose the candidate solution that minimizes the objective, and then perform a second, independent out-of-sample test with the chosen candidate solution.

Because the pickup nodes $\mathcal{N}'_P$ in the spatial network $\mathcal{L}$ represent zones of the planning region, it is likely that there is overlap in the physical locations represented by a pickup node and a vehicle origin node. Volunteers in the ridesharing program will likely be recruited from the community evacuating, and emergency vehicles will likely be located within the community as well. Thus, one mild assumption that we can apply is that each vehicle $v \in V$ has an origin node o_v such that distance $d_{o_v i} = 0$ for exactly one $i \in \mathcal{N}'_P$. We say vehicle v *originates* in zone i, and denote the set of volunteer vehicles that originate in zone i as S_i.

Under this assumption, we provide the details of a greedy heuristic in Algorithm 1 as follows. It constructs a worst-case spatial scenario ω_{spatial} from a Monte Carlo sample Ω_m of random variable $\tilde{\omega}$, defining spatial demand at location $i \in \mathcal{N}'_P$ in ω_{spatial} to be $a^i_{\max}$, the largest total number of evacuees who arrive at location i over all scenarios $\omega \in \Omega_m$. Then, we find a set of vehicles with the capacity to accommodate all demand in ω_{spatial}. For simplicity of language, for the remainder of Section 2.3, "demand" and "evacuees" both refer to the demand in this worst-case scenario.

We assume that in the worst-case scenario, emergency vehicles will be filled to their full capacity. Thus, the overall goal of Algorithm 1 is to hire enough volunteer vehicles to satisfy the difference between the demand and the total capacity of all emergency vehicles. The algorithm first matches demand and volunteer vehicles located at the same node $i \in \mathcal{N}'_P$, hiring and completely filling volunteer vehicles $s \in S_i$ one at a

Algorithm 1 A greedy heuristic for obtaining shared-mobility-based evacuation plans.

1: Define $a^i_{\max} := \max_{\omega \in \Omega_m} \{\sum_{t=1}^{T} a^t_i(\omega)\}$, $\forall\ i \in \mathcal{N}'_P$.
2: Initialize $a^i_{\text{rem}} := a^i_{\max}$, $\forall\ i \in \mathcal{N}'_P$, and $a_{\text{rem}} := \sum_{i \in \mathcal{N}'_P} a^i_{\text{rem}}$.
3: Initialize $S_H := \varnothing$ and $\mathcal{Q}^{\text{rem}}_s := \mathcal{Q}_s$, $\forall\ s \in V$.
4: **for** $i \in \mathcal{N}'_P$ **do**
5: Initialize $E_i := S_i$.
6: **while** $a_{\text{rem}} - \sum_{s \in S_H} \mathcal{Q}^{\text{rem}}_s > \sum_{r \in R} \mathcal{Q}_r$ **and** $a^i_{\text{rem}} > 0$ **and** $|E_i| > 0$ **do**
7: Choose $s \in E_i$ with maximum $\mathcal{Q}^{\text{rem}}_s$. Add s to S_H and delete s from E_i.
8: **if** $\mathcal{Q}^{\text{rem}}_s \leq a^i_{\text{rem}}$ **then**
9: Update $\mathcal{Q}^{\text{rem}}_s := 0$, $a^i_{\text{rem}} := a^i_{\text{rem}} - \mathcal{Q}^{\text{rem}}_s$, and $a_{\text{rem}} := a_{\text{rem}} - \mathcal{Q}^{\text{rem}}_s$.
10: **else**
11: Update $\mathcal{Q}^{\text{rem}}_s := \mathcal{Q}^{\text{rem}}_s - a^i_{\text{rem}}$, $a^i_{\text{rem}} := 0$, and $a_{\text{rem}} := a_{\text{rem}} - a^i_{\text{rem}}$.
12: **end if**
13: **end while**
14: **end for**
15: **while** $a_{\text{rem}} - \sum_{s \in S_H} \mathcal{Q}^{\text{rem}}_s > \sum_{r \in R} \mathcal{Q}_r$ **and** $(\sum_{i \in \mathcal{N}'_P} |E_i| > 0$ **or** $\sum_{s \in S_H} \mathcal{Q}^{\text{rem}}_s > 0)$ **do**
16: Choose pickup location i^* with greatest remaining demand $a^{i^*}_{\text{rem}}$, over nodes in $\mathcal{N}'_P$.
17: **if** $\sum_{s \in S_H} \mathcal{Q}^{\text{rem}}_s > 0$ **then**
18: Find node $i \in \mathcal{N}'_P$ closest to i^* such that $\mathcal{Q}^{\text{rem}}_s > 0$ for $s \in S_i$. Repeat Steps 8-12.
19: **else**
20: Find node $i \in \mathcal{N}'_P$ closest to i^* such that $|E_i| > 0$. Repeat Steps 7-12.
21: **end if**
22: **end while**
23: **return** S_H.

time, from the largest to the smallest vehicle capacity $\mathcal{Q}_s$ for $s \in S_i$. It continues to hire volunteer vehicles and fill them with demand at node $i \in \mathcal{N}'_P$ until the location runs out of volunteer vehicles or people who need rides, or until a global condition is met, i.e., the remaining unassigned demand from all locations is less than or equal to total capacity of emergency vehicles. To do this, a set of empty volunteer vehicles, E_i, is initialized as S_i. Each volunteer vehicle s has a remaining capacity variable $\mathcal{Q}^{\text{rem}}_s$, which is initialized as its capacity $\mathcal{Q}_s$. When a volunteer vehicle $s \in E_i$ is hired, it is removed from E_i and added to set S_H. Once a vehicle is hired, it is filled to capacity, or with remaining unassigned individuals at the node, whichever is smaller.

At this point, we simultaneously decrease $\mathcal{Q}_s^{\text{rem}}$ and a_{rem}^i, a variable that represents remaining unassigned demand at location $i \in \mathcal{N}_P'$, accordingly.

Once we have matched as many people and seats as possible at each location $i \in \mathcal{N}_P'$, we test to see if the total remaining demand for all locations $i \in \mathcal{N}_P'$, denoted a_{rem}, exceeds the total capacity of all emergency vehicles $r \in R$. If so, we stop. Otherwise, we continue to fill and hire volunteer vehicles until remaining demand a_{rem} falls to the level of the total capacity of the emergency vehicles or we run out of empty volunteer vehicles in $\bigcup_{i \in \mathcal{N}_P'} E_i$ and remaining capacity in hired vehicles. To do this, Algorithm 1 repeatedly chooses the pickup location i^* with the greatest remaining demand, $i^* = \arg\max_{i \in \mathcal{N}_P'} a_{\text{rem}}^i$. If there is a volunteer vehicle s in the set of hired cars S_H such that remaining capacity $\mathcal{Q}_s^{\text{rem}} > 0$, the algorithm identifies the closest such vehicle to i^*, and fills the vehicle as much as possible. Otherwise, the closest empty volunteer vehicle, in the set $E_j \subseteq \bigcup_{i \in \mathcal{N}_P'} E_i$, to i^* is added to S_H and removed from E_j, and filled as much as possible.

Specifically, in Step 1 of Algorithm 1, we construct a worst case spatial demand scenario, by examining total demand over periods $1, \ldots, T$ at pickup location $i \in \mathcal{N}_P'$ and maximizing this value over all scenarios $\omega \in \Omega_m$. The sample size $|\Omega_m|$ only affects this first step – thus, Algorithm 1 can be used with large Monte Carlo samples.

We initialize a set of variables in Steps 2–3. The cars that have been assigned at least one evacuee are contained in the set S_H, and thus, S_H is empty at first. Steps 4–14 hire the initial set of volunteer vehicles, matching demand and vehicles by location. Step 5 initializes a set of cars that have not yet been assigned evacuees, E_i, as S_i, the set of volunteer vehicles that originate at i. Then the condition in Step 6 checks to see if the total remaining demand is greater than the sum of the capacities of the emergency vehicles and the remaining capacity of the volunteer vehicles that have already been hired. The while loop also checks to see if there is still remaining demand in the location and cars to be hired in the location. If these conditions are

met, in Step 7, we hire the volunteer vehicle at location i with greatest capacity. Then, in Step 8, we check to see if demand left at the location is at least as large as the capacity of the volunteer vehicle we have hired. In this case, in Step 9, we fill the volunteer vehicle and decrease the remaining demand. Otherwise, remaining capacity in the volunteer vehicle exceeds demand and we have seats left in the vehicle, as seen in Step 11. The "for" loop can be executed in a particular order based on the specifics of the instance. For example, if a network has a single emergency origin location, pickup nodes can be chosen in order from furthest to closest to the emergency origin node, so that if the total remaining demand condition in Step 6 is met quickly, the areas furthest from the emergency vehicles will be assigned volunteer vehicles.

Then, Steps 15–22 fill all remaining capacity in the hired cars, and add more cars if necessary. The loop runs as long as total remaining demand exceeds the total emergency car capacity and either some car has not been hired or some capacity remains in a hired car. In Step 16, we choose the location with the most demand that has not yet been assigned a volunteer vehicle. Then, Step 17 checks to see if any of the hired volunteer vehicles are partially full. If so, in Step 18, demand at location i^* is fulfilled by the closest volunteer vehicle. Note that there is at most one hired volunteer vehicle s at location i such that $\mathcal{Q}_s^{\text{rem}} > 0$ at any given time. If all hired cars are full, the closest available car to location i^* is hired and filled in Step 20.

2.4 Generation and Parameter Settings of Test Instances

To test our models and Algorithm 1, we generate a set of test cases of diverse sizes based on the greater Charleston area, a region located on the Atlantic coast that experienced a mandatory evacuation order in 2018 for Hurricane Florence (see *Manno*, 2018). Specifically, we consider instances based on five regional networks and seven hour-long time periods the day of the evacuation. We generate demand samples, volunteer and emergency vehicle sets using census data and traffic count

data from the day of the evacuation, which we describe in detail in this section. We also illustrate the demand and supply heat maps for the five regions in Figures 2.4–2.6. Each instance has finite horizon $T = 4$, corresponding to 15-minute intervals during each hour studied.

2.4.1 Scenario generation method overview

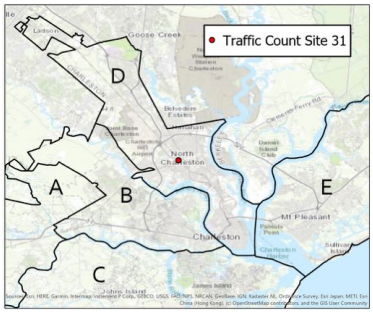
In the literature, random evacuation scenarios are generated in a variety of ways. *Legg et al.* (2010) represent regional hurricane hazard in North Carolina with a set of 100 hurricane scenarios (with 33 involving evacuation) constructed using HAZUS-MH probabilistic analysis (*FEMA*, 2006). *Li et al.* (2012) use these 33 scenarios to study a shelter location model in North Carolina, using estimates of geographical distribution and volume of evacuees and individuals requiring shelter determined by HAZUS-MH software. They also describe temporal distribution of evacuation using the Rayleigh distribution, which was used by *Tweedie et al.* (1986) to describe estimates for evacuation times given by Oklahoma Civil Defense Office experts. *Ben-Tal et al.* (2011), who study a dynamic traffic assignment problem in emergency evacution, assume the temporal distribution of evacuation is an S-shaped curve and utilize a mathematical formulation given by *Radwan et al.* (1985).

To generate demand scenarios in different samples, we construct a set of five evacuation networks, based on Charleston County, South Carolina. We use hourly traffic counts from a continuous count station on Highway I-26 in Charleston on the day the 2018 Hurricane Florence evacuation order took effect, to estimate of the temporal distribution of demand. Because the populations who can request service from our program are distinct from populations who are most likely to provide service, we use census data on the demographics of tracts in Charleston County to estimate the geographical distribution of both evacuees requesting service and volunteers available.

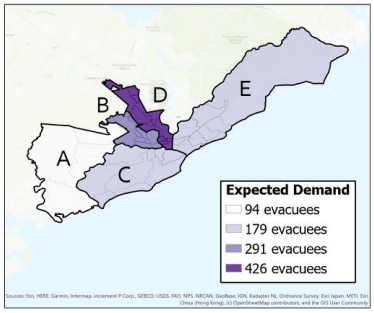
2.4.2 Parameter settings

2.4.2.1 Building spatiotemporal networks

To construct a spatial network, census tracts in Charleston County are aggregated to obtain 48 spatial zones – each represented as a node in the network. Distances and travel times between nodes are determined using the centers of the spatial zones. The coordinates of each geographic center are first found using ArcGIS Pro, and then utilized in Google Distance API to determine d_{ij} and t_{ij} for each $i,\ j \in \mathcal{N}'$. This spatial network is separated into five subnetworks, ranging in node size from 4 to 16, based on natural geographical boundaries, as depicted in Figure 2.2a.



(a) Location of traffic count Site 31 and demarcation of regions in Charleston County.



(b) Total expected demand for ridesharing services in each region.

Figure 2.2: Illustration of regions and demand hot zones in Charleston County.

Temporal evacuation patterns are estimated using hourly traffic count data from Site 31, a Continuous Count Station on Highway I-26 in Charleston, on September 11, 2018, from 6:01 am-1:00 pm (*South Carolina Department of Transportation*, 2018). The location of Site 31 also can be seen in Figure 2.2a. The governor of South Carolina had issued a mandatory evacuation order for coastal areas by noon of that day (*Manno*, 2018). We consider each hourlong time period as a separate test case, with $T = 4$ intervals of 15 minutes each.

A spatiotemporal network $(\mathcal{N}, \mathcal{A})$ is constructed from each regional subnetwork $(\mathcal{N}', \mathcal{A}')$, with origin node o_i^0 for each spatial zone $i \in \mathcal{N}'$ and origin node o_{j*}^0 for a

centrally located spatial zone. Here, we assume that for each volunteer s, we have $o_s^0 := o_i^0$, the zone i where the volunteer lives. For each emergency vehicle r, we have $o_r^0 := o_{j^*}^0$, a depot where all emergency vehicles originate. In this way, regardless of the number of vehicles utilized in a particular test, the structure of the spatiotemporal network remains the same. In addition to origin nodes, the spatiotemporal network includes pickup nodes i^t for $i \in \mathcal{N}'$, $t = 1, \ldots, T$ and safe-zone node h^{T+1}. An arc is drawn from i^t to $j^{t'}$ if $t' - t = \left\lceil \frac{t_{ij}}{900} \right\rceil$. For every spatial zone, a state-determined evacuation route run through either the zone or an adjacent zone. Thus, the safe zone is reachable in one period for every pickup node.

2.4.2.2 Demand generation and samples

Demand at each spatiotemporal node is assumed to be Normally distributed with mean $\mu_{i,t}$ and variance $\sigma_{i,t}^2$, independent of demand at other times or locations. The mean $\mu_{i,t}$ is determined by first estimating total demand at location $i \in \mathcal{N}_P'$ over all periods, denoted μ_i, using data on the median income and number of elderly people living in census tracts located in each node (*United States Census Bureau*, 2018a,b). The mean value of elderly people (65+) assumed to take advantage of our service is determined by taking a percentage of the total elderly population in that census tract. The percentage is determined by the quartile of the median income for people 65 years and older in that census tract, as seen in Table 2.3. Here, median income serves as a proxy for vehicle ownership. We then determine μ_i for each zone i by adding the mean value for each tract in the aggregated zone i, and round to the nearest integer. We visualize the spatial distribution of $\mu_i,\ i \in \mathcal{N}'$ for each of the five regions in Figures 2.4a, 2.4c, 2.5a, 2.5c, and 2.6a.

We assume that the same temporal evacuation pattern holds for each location. Using the aforementioned traffic count data, we estimate the percentage of overall evacuees who leave in hour ζ, denoted p_ζ, where $\sum_{\zeta=1}^{7} p_\zeta = 1$. Site 31 is chosen

Table 2.3: Percentage of 65+ population requesting ridesharing-based evacuation service in census tract

Quartile	Median Income Range	Percentage
1	[$16,726, $34.235.50]	5%
2	($34.235.50, $45,600]	2.5%
3	($45,600, $62,106.50]	1.25%
4	($62,106.50, $146,477]	.625%

because the traffic count pattern clearly changed the day of the evacuation deadline. Figure 2.3 shows the similar traffic patterns present on normal work days – Tuesday, September 4 to Friday, September 7, along with Monday, September 10 – and present on weekend/holiday days – the Saturdays, Sundays, and Labor Day Monday. Then, a markedly different pattern appears Tuesday, September 11. Residents were ordered to evacuate by noon that day (*Manno*, 2018).



Figure 2.3: Hourly traffic counts for September 1 to September 13, 2018 on Highway I-26 in Charleston County.

In addition, the data shows a clear pattern of greater westbound traffic the day of the evacuation deadline, in the direction away from the coast, which suggests that many of those passing by this traffic count site were evacuees. While the ratio of total westbound to total eastbound traffic on Tuesday, September 4 is 0.98, meaning traffic is roughly evenly split by east/west, the ratio for westbound to eastbound traffic on Tuesday, September 11 is 2.33. There is an overall reduction in traffic from September 4 to September 11 with about 20% of the total eastbound traffic and about 50% of

the total westbound traffic. Because those traveling in the eastern direction are most likely not immediately evacuating, we use these travelers as a proxy for the proportion of people in the westbound direction who are not evacuees.

For each hourlong period ζ, we calculate the number of evacuees with the following formula: $W_{11}^{\zeta} - W_4^{\zeta} \times (E_{11}^{\zeta}/E_4^{\zeta})$, where W_{11}^{ζ} (or E_{11}^{ζ}) represents the westbound (or eastbound, respectively) traffic count at hour h on September 11, 2018, and W_4^{ζ} (or E_4^{ζ}) represents the westbound (or eastbound, respectively) traffic count at hour h on September 4, 2018, a normal workday. Here, W_4^{ζ} estimates the number of westbound commuters on a normal Tuesday, and $\frac{E_{11}^{\zeta}}{E_4^{\zeta}}$ estimates the proportion of Tuesday westbound commuters who went to work the morning of September 11, 2018 rather than evacuating at that time. The idea is that the eastbound and westbound directions each have regular morning commuters (i.e., two distinct groups that travel in the same direction in the morning and at night). We assume these groups are similar in their evacuation timing decisions. Thus, the proportion of westbound travelers who have decided not to evacuate and are carrying about their normal business on September 11, 2018 should be a similar proportion to the eastbound travelers, all of whom we assume are not evacuating, since they travel toward the coast.

The hours of 6:01 am to 1 pm are found to have the highest volume of evacuees. The percentage of evacuees in each hour of the roughly 2,300 estimated evacuees over the 7 hours is used to estimate p_{ζ}. The parameters p_{ζ} that we use are presented in Table 2.4. For epoch t in period ζ, the mean $\mu_{i,t}$ is determined by multiplying $\frac{p_{\zeta}}{4}\mu_i$. The variance $\sigma_{i,t}^2$ is assumed to be $.3\mu_{i,t}$.

In order to compare the performance of (Exp-SAA) and Algorithm 1 when preferred proportion of evacuation $\gamma = 1$, we construct Monte Carlo samples for smallest network, Region A, and largest network, Region D, for all six time periods with unique demand distributions, 1, 2, 3, 5, 6, and 7, as well as the busiest time period, Period

Table 2.4: Temporal distribution of evacuees at Site 31 during different hours on September 11, 2018.

Period	Percentage of Evacuees
6:01-7:00 am	12%
7:01-8:00 am	18%
8:01-9:00 am	14.5%
9:01-10:00 am	14.5%
10:01-11:00 am	15.5%
11:01 am-12:00 pm	13%
12:01-1:00 pm	12.5%

2, for all five networks. We perform Monte Carlo sampling, generating 5 samples of 50 scenarios each.

To compare the performance of (Exp-SAA), (CC-SAA), and (CC-CVaR-SAA) when desired proportion of evacuation is $\gamma = 0.9$, we generate 5 samples of 100 scenarios each for medium-sized networks Region B, C, and E, for second-busiest time period, Period 5. These samples are reduced to 20 scenarios each, using techniques described in Section 2.5.2. We denote these samples $\Omega_m^{\mathcal{N}'\zeta}$, $m = 1, \ldots, 5$, each of which represents an in-sample test corresponding to network $\mathcal{N}'$ and period ζ. For each scenario $\omega \in \Omega_m^{\mathcal{N}'\zeta}$ and spatiotemporal node $i^t \in \mathcal{N}_P$, the number of arrivals is $a_i^t(\omega) = \max\{[X_{i,t}], 0\}$, where $X_{i,t}$ is a random variable with distribution $N(\mu_{i,t}, \sigma_{i,t}^2)$. For all test instances, two independent samples of 500 scenarios each are generated in the same manner for out-of-sample testing. For ease of notation, we denote the total expected demand over an hour h in all zones in network $\mathcal{N}'$ as $\mu_\zeta^{\mathcal{N}'}$, where $\mu_\zeta^{\mathcal{N}'} = \sum_{i \in \mathcal{N}'} \sum_{t=4(\zeta-1)+1}^{4\zeta} \mu_{i,t}$. These values can be found for each network and period in Table 2.5.

We assume that about 60% of demand will be satisfied by volunteer cars, and 40% by emergency cars. To determine the number of vehicles available in hour h, we use the following formula: $|S| = 0.6 \times \sum_{i \in \mathcal{N}'} \sum_{t=4(\zeta-1)+1}^{4\zeta} \frac{\mu_{i,t}}{\mathcal{Q}_s} \times 2$ and $|R| = 0.4 \times \sum_{i \in \mathcal{N}'} \sum_{t=4(\zeta-1)+1}^{4\zeta} \frac{\mu_{i,t}}{\mathcal{Q}_r} \times 2$. Here, $\mathcal{Q}_v$ is fixed for $v \in R$ and $v \in S$, and 2 is a factor to increase the likelihood that the pool of available cars can handle demand

Table 2.5: Total expected demand for each region and period.

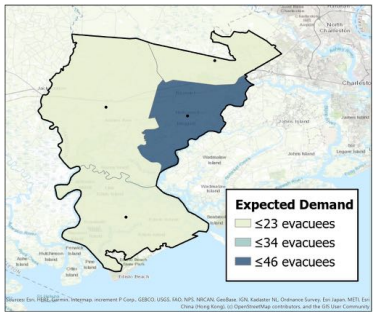
Region	$\lvert\mathcal{N}_P\rvert$	ζ	$\mu_\zeta^{\mathcal{N}'}$
A	4	1	11.280
		2	16.920
		3	13.630
		5	14.570
		6	12.220
		7	11.750
B	10	1	34.920
		2	52.380
		3	42.195
		5	45.105
		6	37.830
		7	36.375
C	9	1	21.480
		2	32.220
		3	25.955
		5	27.745
		6	23.270
		7	22.375
D	16	1	51.120
		2	76.680
		3	61.770
		5	66.030
		6	55.380
		7	53.250
E	9	1	21.480
		2	32.220
		3	25.955
		5	27.745
		6	23.270
		7	22.375

scenarios that arise. The capacities used for this study are $\mathcal{Q}_s = 3$ and $\mathcal{Q}_r = 7$.

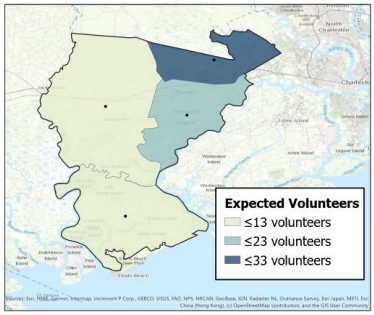
2.4.2.3 Settings of supply (volunteers)

The spatial distribution of volunteer cars is estimated using data on the population size of individuals ages 25-64 and median income in the census tracts at particular nodes (*United States Census Bureau*, 2018a,b). Higher median income in a particular

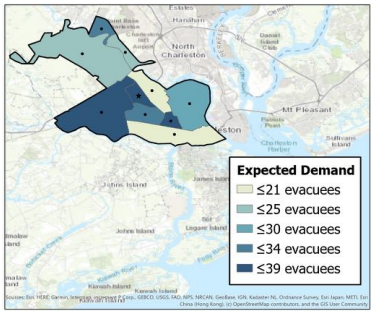
census tract is assumed to increase likelihood of vehicle ownership, and thus, ability to act as a volunteer driver. Each origin node, at location i, is assigned an estimated proportion of volunteer drivers, v_i. The number of volunteer cars at an origin node at location i is determined by calculating $[v_i \cdot |S|]$, the nearest integer. Any vehicles not assigned by this process are randomly assigned to an origin node. The spatial distribution of volunteers on average can be seen for each of the five networks in Figures 2.4b, 2.4d, 2.5b, 2.5d, and 2.6b.



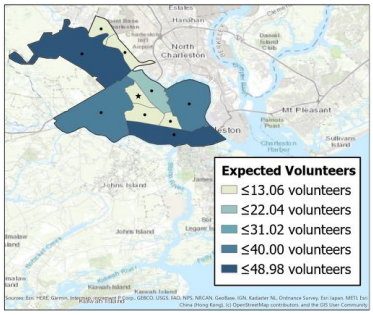
(a) Region A – Expected Demand



(b) Region A – Expected Volunteers



(c) Region B – Expected Demand

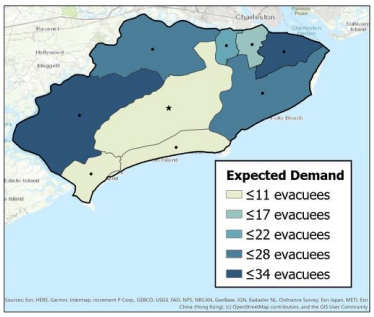


(d) Region B – Expected Volunteers

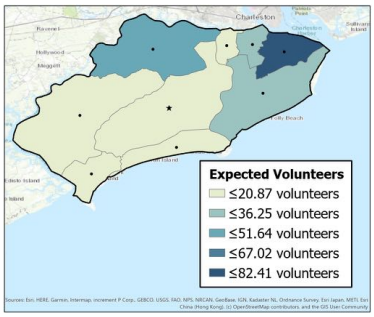
Figure 2.4: Spatial distributions of expected evacuation demand and number of volunteers in Regions A and B.

2.5 Computational Studies

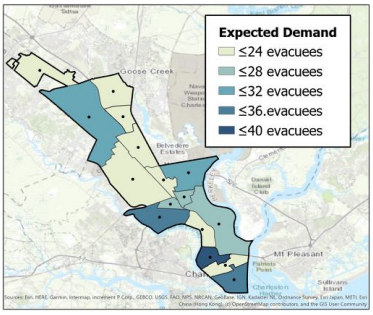
We test our solution methods using the instances developed in Section 2.4. The size and computational difficulty of solving a problem instance is influenced by a few factors. The corresponding network determines the number of spatial pickup nodes, $|\mathcal{N}'_P|$. The network and considered time period determine the number of volunteer



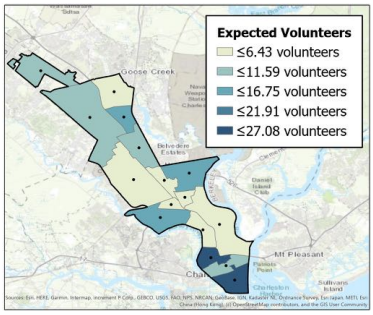
(a) Region C – Expected Demand



(b) Region C – Expected Volunteers

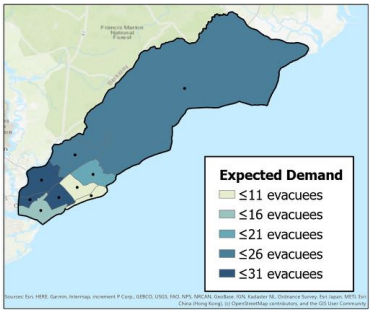


(c) Region D – Expected Demand

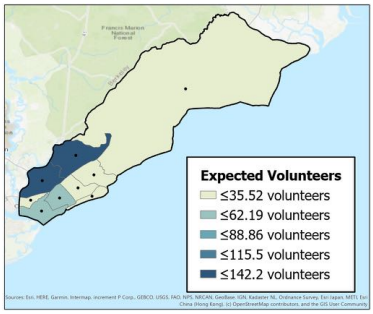


(d) Region D – Expected Volunteers

Figure 2.5: Spatial distributions of expected evacuation demand and number of volunteers in Regions C and D.



(a) Region E – Expected Demand



(b) Region E – Expected Volunteers

Figure 2.6: Spatial distributions of expected evacuation demand and number of volunteers in Region E.

vehicles and emergency vehicles available, as well as their spatial dispersion and the expected number of evacuees requesting service.

In Section 2.5.1, we examine the case when desired proportion of evacuation $\gamma = 1$, comparing the performance of (Exp-SAA) and Algorithm 1. To compare results across different network types, we consider the five networks during the busiest hour during evacuation. To examine the differences within a network for time periods with

different expected period demand, we consider the largest and smallest networks for six unique time periods. As a result, the (Exp-SAA) model and Algorithm 1 are applied to 15 instances. Algorithm 1 is implemented in Python 3.6, while (Exp-SAA) is directly solved by Gurobi 9.5. We utilize relaxation and a scenario-decomposition procedure to find an initial feasible solution for (Exp-SAA), to improve solution quality, that we describe further in Section 2.5.1.

In Section 2.5.2, we compare (Exp-SAA), (CC-SAA), and (CC-CVaR-SAA), with $\gamma = 0.9$, using three test instances from three medium-sized networks. All models are solved directly by Gurobi 9.5. We reduce the size of the in-sample scenario sets by choosing a representative subset of the Monte Carlo samples, described further in Section 2.5.2.

All the tests were performed on a desktop computer with an Intel Core E5-2630 v4 CPU 2.20 GHz and 128 GB of RAM. For each instance tested for each of the three stochastic optimization formulations, five independent samples are used to obtain candidate solutions, $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_5$.

2.5.1 Comparison of (Exp-SAA) and heuristic for $\gamma = 1$

We first consider the case when emergency planners want to serve all demand, whenever possible. In this setting, solving (Exp-SAA) with high penalty cost and Algorithm 1 with its conservative solutions are most appropriate. We compare the performance of the two approaches with 15 test cases of various sizes. When deciding the first-stage recruitment cost, our goal is to balance the number of volunteers hired and expected distance traveled. We set recruitment cost $\mathbf{f}^T\mathbf{x} = 1/(2|S|)\sum_{s\in S} x_s$ and operational phase objective $Q(\mathbf{x}, \mathbf{a}(\omega)) = \frac{1}{20(|R|+|S|)}\sum_{v\in V}\sum_{(i^t,j^{t'})\in A} y_{vi^tj^{t'}\omega}d_{ij} + 50000c_\omega$. Here, we normalize the number of volunteers hired by dividing by those available, as well as the total expected distance traveled by the number of cars available and minimum distance traveled by any hired car to the safe zone. We also choose a high

penalty to ensure that the evacuation threshold is satisfied whenever possible.

To perform the computationally expensive tests of optimizing (Exp-SAA) on regional subnetworks with 16 nodes with a tight quality-of-service constraint, we first find an initial feasible solution. To do so, we relax the second stage decision variables, $\mathbf{y}_\omega$ and $\mathbf{q}_\omega$, keeping only the first stage decision variables $\mathbf{x}$ and emergency hiring decision variables $\mathbf{z}_\omega$ integer, in order to determine potential feasible solution $\tilde{\mathbf{x}}$ and corresponding hired volunteer vehicles $\tilde{S}$. Then, we ensure the solution is truly feasible by solving the following integer program to obtain an initial feasible solution $\tilde{\mathbf{x}}'$.

$$\begin{aligned} &\min \mathbf{f}^T\mathbf{x} && \text{(InitFeas)}\\ &\text{s.t. } x_s = 1, \quad s \in \tilde{S}\\ &\qquad c_\omega = 0, \quad \omega \in \Omega_m^{\mathcal{N}'\zeta}\\ &\qquad \text{(2.1b)–(2.1l)}. \end{aligned}$$

This ensures that the initial feasible solution $\tilde{\mathbf{x}}'$ can serve all scenarios. We provide initial feasible values for the other decision variables by solving the operational phase problem, assuming all demand is served, i.e., $Q(\tilde{\mathbf{x}}', \mathbf{a}(\omega))|_{c_\omega=0}$, for each in-sample scenario $\omega \in \Omega_m^{\mathcal{N}'\zeta}$. In cases in which solving Model (InitFeas) does not produce a feasible solution within the time limit, we solve $Q(\tilde{\mathbf{x}}, \mathbf{a}(\omega))|_{c_\omega=0}$ for each in-sample scenario using the relaxation solution, obtaining initial feasible values for any scenarios that the relaxation solution can satisfy the quality of service constraint. (This is used to allow Gurobi to construct a full initial feasible solution during the in-sample test.) We call these the final phase initial feasible tests.

To compare the computational time and effectiveness of Algorithm 1 directly with (Exp-SAA), we use the same five Monte Carlo samples with 50 scenarios and two independent out-of-sample tests with 500 scenarios each. We use the same parameters

Table 2.6: Gurobi parameter settings for solving (Exp-SAA) versus Algorithm 1 comparison

Method	Test	MIP-Gap	Time Limit (s)	MIP-Focus
(Exp-SAA)	Relaxation	0.05	3600	Default
	InitFeas	0.05	3600	1*
	Final Phase Initial Feasible	0.10	300	1*
	In Sample	0.01	3600	Default
Both	All Candidate Solutions (Out of Sample)	0.10	300	1*
	Final Chosen Solution (Out of Sample)	0.10	300	1*

* Directs Gurobi to focus on finding feasible solutions.

for out-of-sample tests for (Exp-SAA) and Algorithm 1, listed in Table 2.6. Descriptions of the problem sizes can be found in Table 2.7, as well as in Table 2.8. We include Column '# Vol Zones' in the result tables to indicate the number of unique zone locations where volunteers originate.

Table 2.7: Problem instance sizes during the busiest period for in-sample tests for (Exp-SAA), including total expected network period demand, $\mu_2^{\mathcal{N}'}$.

Region	$\lvert\mathcal{N}'_P\rvert$	$\mu_2^{\mathcal{N}'}$	$\lvert S\rvert$	# Vol Zones	Integer Variables	# of Constraints
A	4	16.92	6	2	18356	14454
C	9	32.22	12	5	103012	56307
E	9	32.22	12	6	126312	58506
B	10	52.38	20	7	299670	107463
D	16	76.68	30	13	993330	248917

Table 2.8: Problem instance sizes for smallest and largest networks, Region A and D, for in-sample tests for (Exp-SAA), including total expected network period demand, $\mu_\zeta^{\mathcal{N}'}$.

Region	$\lvert\mathcal{N}'_P\rvert$	Period	$\mu_\zeta^{\mathcal{N}'}$	$\lvert S\rvert$	# Vol Zones	Integer Variables	Constraints
A	4	1	11.28	4	2	11404	9402
		7	11.75	4	2	11404	9402
		6	12.22	5	2	13755	11103
		3	13.63	5	2	16155	12803
		5	14.57	5	2	16155	12803
		2	16.92	6	2	18356	14454
D	16	1	51.12	20	11	660120	166959
		7	53.25	21	11	687221	173360
		6	55.38	23	12	733573	185661
		3	61.77	25	12	807125	204363
		5	66.03	26	12	861476	217164
		2	76.68	30	13	993330	248917

For all instances, optimal solutions to (Exp-SAA) hire the same or fewer volunteers than the heuristic. Solving (Exp-SAA) also ensures full evacuation with probability at least 0.99 for all instances. In contrast, Algorithm 1 ensures full evacuation with probability 1 for all tests, with more volunteers required for 12 of 15 instances. It can find candidate solutions in less than a second for all tests.

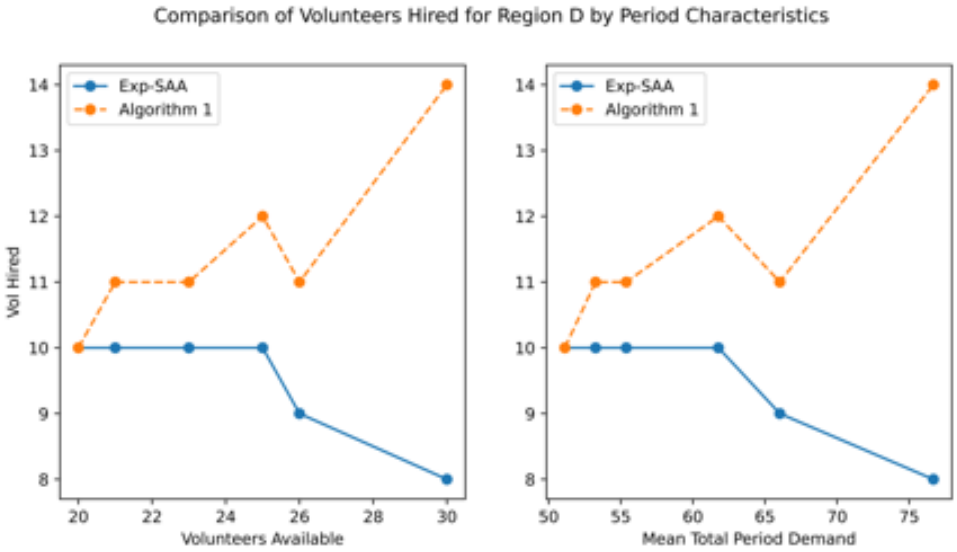


Figure 2.7: Total number of volunteers hired for Region D for six test periods.

Figure 2.7 depicts the number of volunteers hired by the two methods for the largest network, Region D, across the six test periods. In the lefthand figure, we see that (Exp-SAA) is able to take advantage of increasing number of volunteers available and find solutions with lower recruitment cost, much more than Algorithm 1. The righthand figure illustrates how increasing total expected demand can yield higher recruitment cost for the heuristic, while solving (Exp-SAA) does not show the same trend.

In Table 2.9, we compare the in-sample computational runtime (RT) and out-of-sample objective values for the five networks during the busiest period. The rows of Table 2.9 are ordered in increasing problem size, in terms of pickup nodes, total expected demand per period, volunteers available, and unique volunteer locations.

Table 2.9: In-sample computational runtime (in second) and out-of-sample objective values for (Exp-SAA) and Algorithm 1 for the busiest period, Period 2.

Region	(Exp-SAA) In Sample		Algorithm 1 In Sample	Objective Value	
	Mean InitFeas RT	Mean Model RT	Mean RT	(Exp-SAA)	Algorithm 1
A	19.0	2443.5	*	0.861	0.958
C	393.8	12.7%**	*	3.889	0.695
E	633.7	8.4%**	*	0.702	0.754
B	3023.9	28.6%**	*	2.107	0.624
D	7714.4	29.1%**	*	0.504	0.595

* indicates that the value is less than one second.

** indicates that all samples time out at 3600 s, and mean optimality gap is given.

Table 2.10: In-sample computational runtime (in second) and out-of-sample objective values for (Exp-SAA) and Algorithm 1 for all tests.

Region	Period	(Exp-SAA) In Sample		Algorithm 1 In Sample	Obj Val	
		Mean Init Feas RT	Mean Model RT	Mean RT	(Exp-SAA)	Algorithm 1
A	1	4.3	63.0	*	1.244	1.245
	7	8.8	125.8	*	1.306	1.523
	6	7.0	72.6	*	35.637	1.153
	3	13.8	556.5	*	0.818	0.818
	5	13.8	856.6	*	0.849	0.994
	2	19.0	2443.5	*	0.861	0.958
D	1	6587.7	22.0%**	*	0.722	0.734
	7	8607.3	37.2%**	*	0.699	0.722
	6	8675.2	26.8%**	*	0.66	0.672
	3	8371.3	26.3%**	*	0.605	0.649
	5	8021.3	41.0%**	*	0.563	0.603
	2	7714.4	29.1%**	*	0.504	0.595

* indicates that the value is less than one second.

** indicates that all samples time out at 3600 s, and mean optimality gap is given.

Note that the mean total runtime to construct the initial feasible solution is less than an hour for the four smaller regions, and a few hours for the largest region. Using these initial feasible solutions, an optimal solution is found for smallest network, Region A, within the hour time limit. The other larger test cases, with number of integer variables ranging from 100,000 to 990,000 time out, with mean optimality gaps ranging from 8.4% to 29.1%. As mentioned earlier, the heuristic finds candidate solutions very quickly. We also include the results for all six unique periods and two test networks (region A and region D) in Table 2.10.

In three of the five cases in Table 2.9, directly solving (Exp-SAA) outperforms Algorithm 1 in terms of out-of-sample objective. In Table 2.11, the gains in the objective value generally come from hiring fewer volunteers. Algorithm 1 outperforms

Table 2.11: Out-of-sample solution performance of expectation-based formulation and Algorithm 1 for the busiest period, Period 2.

Region	Volunteers Hired		Probability of Full Evacuation		Total Distance Traveled	
	(Exp-SAA)	Algorithm 1	(Exp-SAA)	Algorithm 1	(Exp-SAA)	Algorithm 1
A	3	4	1	1	97.7	99.9
C	4	5	0.998	1	159.0	155.9
E	5	6	1	1	157.9	161.4
B	7	10	0.998	1	192.2	194.3
D	8	14	1	1	288.7	281.8

Table 2.12: Out-of-sample solution performance of expectation-based formulation and Algorithm 1 for all tests.

Region	Period	Volunteers Hired		Probability of Full Evacuation		Total Distance Traveled	
		Exp	Algorithm 1	Exp	Algorithm 1	Exp	Algorithm 1
A	1	3	3	1	1	86.9	87.0
	7	3	4	1	1	93.1	102.3
	6	3	4	0.99	1	83.6	90.3
	3	2	2	1	1	86.5	86.6
	5	2	3	1	1	90.9	97.1
	2	3	4	1	1	97.7	99.9
D	1	10	10	1	1	245.6	251.5
	7	10	11	1	1	248.9	248.2
	6	10	11	1	1	256.9	251.3
	3	10	12	1	1	259.2	261.7
	5	9	11	1	1	264.9	266.5
	2	8	14	1	1	288.7	281.8

(Exp-SAA) in the other two cases in Table 2.9 in terms of out-of-sample objective due to violations of the evacuation threshold, in 0.2% of out-of-sample tests. Thus, the more conservative solutions given by the heuristic can be protective against rare, difficult scenarios. On the other hand, (Exp-SAA) can provide lower cost solutions than the heuristic – with as much as 40% reduction in volunteers, as seen in Region D, Period 2. Detailed out-of-sample solution performance results for Regions A and D and six unique time periods can be seen in Table 2.12.

2.5.2 Comparison of different stochastic optimization models

Next, we consider the case where multiple modes of evacuation are available to carless populations, and emergency planners are interested to find a cost-effective volunteer recruitment decision that ensures a high proportion of demand is served,

at a particular reliability level. For (CC-SAA) and (CC-CVaR-SAA), we consider probability and CVaR threshold $1 - \epsilon_0 = 0.80$.

Note that because we consider the CVaR formulation as an approximation of the chance-constrained formulation, we perform the first phase of out-of-sample tests in the same fashion for both – finding the 475 best performing out-of-sample scenarios with $c_\omega = 0$ and rerunning the other 25 scenarios with $c_\omega = \gamma$, as described in Section 2.2.3. We use five samples with 20 scenarios each.

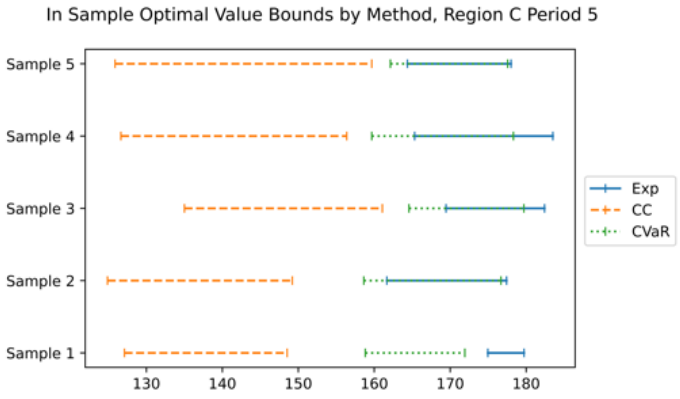
We test the three formulations on medium-sized networks Regions B, C, and E during Period 5. Table 2.13 summarizes the problem instance sizes for each setting combination, in terms of the number of integer variables and the number of constraints in each formulation. We consider the cost function $\mathbf{f}^T\mathbf{x} = 20\sum_{s\in S} x_s$, and set penalty cost $\mathcal{M}_e = 100,000$. During the in-sample tests, to improve computational efficiency, we exclude the expected penalty cost from the expectation-based formulation. (These formulations are equivalent in-sample, because with all scenarios known, the optimal solution to (Exp-SAA) satisfies the quality of service threshold.)

Table 2.13: Problem instance sizes for three methods' in-sample tests, including total expected network period demand, $\mu_5^{\mathcal{N}'}$.

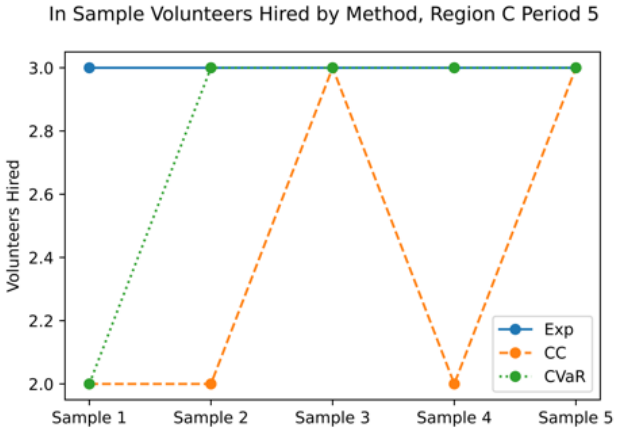
Region	Nodes	$\mu_5^{\mathcal{N}'}$	Integer Variables			Constraints		
			Exp	CC	CVaR	Exp	CC	CVaR
B	10	45.105	105598	105618	105598	38070	38071	38071
C	9	27.745	35831	35851	35831	19745	19746	19746
E	9	27.745	44551	44571	44551	20585	20586	20586

In Figure 2.8, we compare the in-sample performance of the three models for Region C, during Period 5. The time limit is 3600 seconds for solving each instance. Figure 2.8a illustrates that (CC-SAA), with its additional binary variables, has a larger optimality gap than the other two models. There is also a clear difference between the optimality bounds for (CC-SAA) and the other two. Thus, the best solution found by (CC-SAA) is not feasible to the more risk-averse tests for each sample. On the other hand, there is a difference between the optimality gaps of

(Exp-SAA) and (CC-SAA) in Sample 1. In Figure 2.8b, we see that in Sample 1, where smaller optimality gaps are reached for the three approaches and separation emerges, (CC-SAA) yields the same number of hired volunteers as (CC-SAA), fewer than (Exp-SAA). In general, (CC-SAA) yields fewer or the same number of volunteer hires than (Exp-SAA) and (CC-SAA), as expected.



(a)



(b)

Figure 2.8: In-sample performance for three methods for Region C, Period 5.

We see a similar pattern in optimality bounds for Region E, a test instance of approximately the same size as Region C – where there is a clear separation between (Exp-SAA) and (CC-SAA). The three models all produce candidate solutions consisting of three recruited volunteers for each sample in this test case. Region B, the largest test case, has overlapping optimality bounds for all three models. For prob-

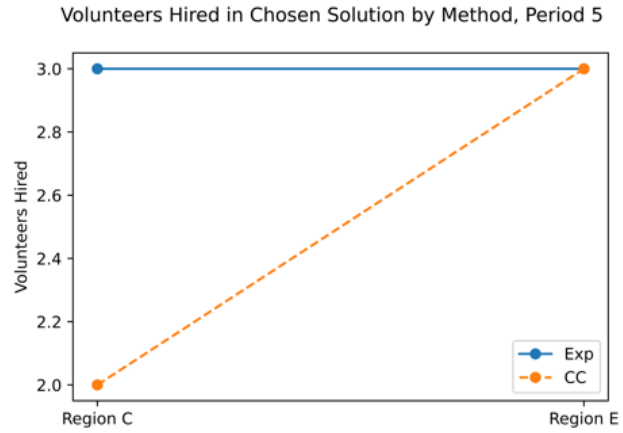
lems of this size, where it is difficult to distinguish between solutions produced by the different stochastic programs, it makes sense to use the simplest computational method – (Exp-SAA).

When comparing out-of-sample performance of the chosen solution for each method, we focus on Regions C and E, and solutions produced by models (Exp-SAA) and (CC-SAA), with their comparison depicted in Figure 2.9. The solution produced by (CC-SAA) has the same or fewer volunteers for each test instance, as seen in Figure 2.9a, with lower or same probability of full evacuation, as seen in Figure 2.9b. The chance-constrained formulation produces a solution that satisfies the probabilistic constraint for both test instances. The expectation-based formulation satisfies the evacuation threshold constraint with probability at least 0.99.

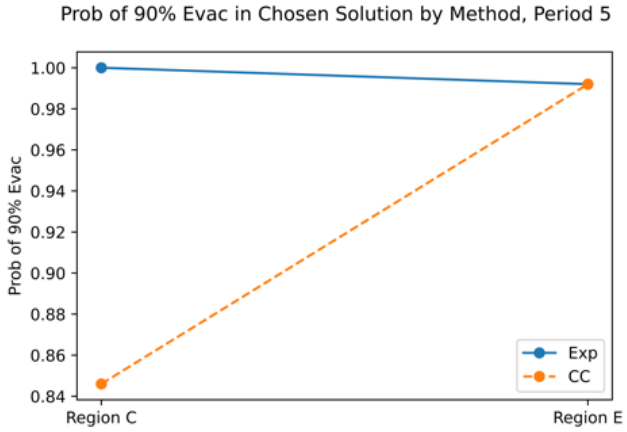
On the other hand, expected total distance traveled by all vehicles is higher for the chance-constrained formulation in both test cases, as shown in Figure 2.9c. The difference is larger for Region C, the case in which fewer vehicles are hired by (CC-SAA). Thus, for smaller problem sizes, with around 35,000 to 45,000 integer variables and 20,000 constraints, both (Exp-SAA) and (CC-SAA) can produce distinct solutions using an off-the-shelf solver. Candidate solutions chosen by the SAA method for each problem can have different benefits and disadvantages – as seen in Region C, where fewer vehicles were hired with lower probability of the evacuation threshold met by (CC-SAA). It is also possible for the methods to produce similar candidate solutions, as seen in Region E.

2.6 Concluding Remarks

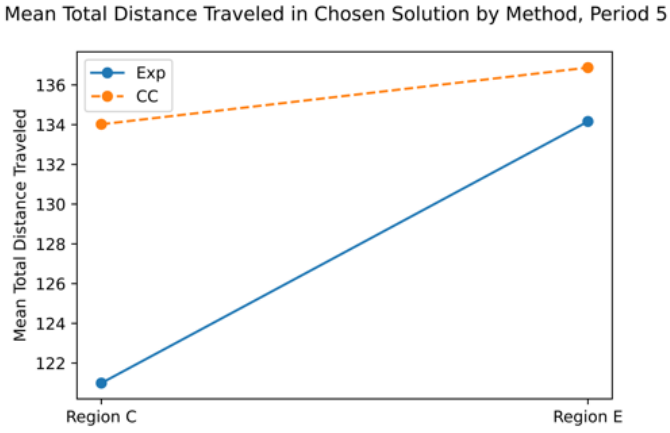
In this chapter, we examined a ridesharing-based evacuation problem. We formulated the problem as a two-stage stochastic program, considering three levels of risk attitudes in terms of unserved evacuees, and developed a heuristic method for obtaining solutions for large-scale problems. We tested our models and the heuristic



(a)



(b)



(c)

Figure 2.9: Chosen solution performance for three methods for Regions C and E, Period 5.

approach using a variety of instances, with different geographical sizes and varying population density, based on the Charleston area, which experienced a mandatory hurricane evacuation in September 2018.

In our numerical studies, the expectation-based formulation produced solutions that ensured probability of full evacuation of at least 0.99, with respect to the out-of-sample distribution. We found that the heuristic algorithm produced solutions that required the same or slightly more volunteers, while satisfying the evacuation threshold in all out-of-sample scenarios. When we considered a 90% evacuation threshold, appropriate for multimodal settings, we discovered that the chance-constrained formulation produced a solution that required fewer volunteers than the expectation-based formulation.

We considered heterogeneous hiring decisions, routing decisions, and loading decisions, under uncertainty in spatiotemporal demand. One interesting direction for future research is to examine the travel-time uncertainty in addition to the evacuation demand uncertainty, and plan evacuation routes as well in the first stage. Further work can also be done to improve computational performance of the three stochastic optimization formulations on larger problem sizes, taking the advantage of special problem structure, and improving the sampling approach for obtaining better quality solutions more efficiently.

CHAPTER III

The Costs of Overcrowding (and Release): Strategic Discharges for Isolated Facilities During Epidemiological Outbreaks

3.1 Introductory Remarks

In this chapter, we consider how to de-densify an isolated facility during an epidemiological outbreak to best mitigate systemic health and release-related risk, as well as develop convenient sequential decision making tools for broader pandemic containment applications. We formulate the problem as a multi-criteria, finite-horizon Markov Decision Process (MDP) that optimizes the release decisions, considering populations with varying health risk levels and release categories (e.g., recidivism risk levels or length remaining in nursing home contract). Recall from Chapter I that we consider two solution approaches – one based on sparse lookahead trees and another, Q learning with linear function approximation.

3.1.1 Methodological Contributions

In addition to developing an MDP model for the de-densification problem and implementing solution methods, we provide three main methodological contributions.

- First, the sparse sampling algorithm has strong theoretical properties but only provides a single action, rather than the policy for an entire trajectory, which makes it difficult to test the performance of the policy on independent out-of-sample trajectories. To mitigate this, we build an algorithm called *Split-Trajectory Sparse Sampling* that uses k-means clustering and multiple calls to the sparse sampling algorithm to evaluate the policy derived by the lookahead tree on a large number of independent samples.

- Second, the theoretical properties promised by the sparse sampling algorithm require sampling fewer child nodes for each node than algorithms that estimate more than one optimal action, but the number of child nodes is still large in a practical sense. For this reason, we develop an adaptation of the baseline sparse sampling algorithm also using k-means clustering. In our computational tests, we find the *Split-Trajectory Sparse Sampling with Clusters* adaptation that uses this version of the sparse sampling algorithm outperforms the *Split-Trajectory Sparse Sampling* method while taking similar computational time. Both of these methodological advances can be applied to other generic generative MDPs.

- Our final methodological contribution is specific to epidemiological MDPs and involves the definition of basis functions for Q learning with linear function approximation. We demonstrate the effectiveness of our basis functions under particular contact rate assumptions and priority weights.

3.1.2 Broader Contributions

In addition to methodological advances, we develop open-source R code, freely available on Github, that acts as a generic framework for generative MDP analysis and solution approaches. Our framework allows the public to define their own generative MDPs, including state, action, and reward components and state-action transition

simulators. Our open-source code includes implementations of the *Split-Trajectory Sparse Sampling* algorithms with and without clusters that we introduce in Section 3.5.1, as well as tools to simulate out-of-sample trajectories that result from applying different types of policies.

We also develop a set of epidemiological test instances, adapted from stochastic individual contact models in the peer-reviewed EpiModel package in R (*Jenness et al.*, 2018) that can be used as benchmark for further methodological development in the solution of multi-objective epidemiological MDPs. Our test instances simulate homogeneous mixing of heterogeneous health risk populations with non-health features (that do not impact infection transmission but impact other aspects of the reward function).

Finally, in our numerical studies, we demonstrate the effectiveness of *Split-Trajectory Sparse Sampling with Clusters* at improving long-term cost over a static benchmark. We note the test instances for which the Q learning with linear function approximation method improves the objective over the static benchmark, as well as the method's shorter relative runtime. We delve into the dynamic policies and resulting cost breakdown derived from the cluster method, demonstrating the ability of the priority weights to yield differentiated policies.

3.2 Literature Review

This work is interdisciplinary, situated in both epidemiological modeling and mathematical optimization fields. In this section, we begin with a review of epidemiological models, including the compartmental framework that we use, the stochastic individual contact model in our test instances, and their application in pandemic settings. Then, we move into a discussion of optimal control models in epidemiology, and review the related work in the control of pandemics. Finally, we discuss the particular framework that we utilize for our decision-making model, MDPs, in the

context of dynamic control, and notable solution approaches previously developed in the literature.

3.2.1 Epidemiological Models

To use our multi-objective, epidemiological MDP framework, the underlying disease model should include a susceptible compartment and heterogeneous health risk levels, as well as a random simulator. The model should assume homogeneous mixing. In our epidemiological test instances, we utilize agent-based stochastic individual contact models (ICMs). Stochastic ICMs are random analogues of deterministic compartmental models (DCMs), benchmark formulations that date back to *Kermack and McKendrick* (1927). DCMs use ordinary differential equations (ODEs) to describe disease spread in a population with homogeneous mixing as a function of a set of epidemiological parameters (*Brauer*, 2008). While DCMs are inherently deterministic, they are used in our MDP by randomly drawing epidemiological parameters from a stochastic distribution each epoch.

Many researchers utilize DCMs to model the spread of the COVID-19 pandemic, fitting outbreak data to curves by estimating epidemiological parameters (*Fanelli and Piazza*, 2020), modeling distinct stages of the pandemic (*Mishra et al.*, 2020), and testing the effectiveness of quarantine (*Hou et al.*, 2020). *Wang et al.* (2020) use deterministic ODEs to describe their model, and then perform stochastic simulations using the τ-leaping method (*Keeling and Rohani*, 2011) to consider the effects of different social distancing measures. Moreover, stochastic ICMs are utilized to model the COVID-19 pandemic by *Churches et al.* (2020), who extend the stochastic ICM in open-source, peer-reviewed EpiModel (*Jenness et al.*, 2018) from SIR to seven compartments, including compartments representing isolation, hospitalization, and death. In this chapter, we also extend the stochastic ICM in EpiModel. However, in line with our objectives, we consider heterogeneous health risk and a simpler generative

model, SIR compartments with homogeneous mixing.

3.2.2 Optimal Control in Epidemiology

While epidemiological models can be utilized to predict disease spread and compare the effects of various interventions, they do not necessarily find optimal courses of action. Work in mathematical optimization aims to bridge this gap, with disease control models that aim to identify strategies to minimize new infections (*Mbah and Gilligan*, 2011) or the exponential decay rate of infections (*Nowzari et al.*, 2015). Two central strains of research in this field are resource allocation models and network interdiction models.

Resource allocation models find optimal epidemiological interventions under the constraint of limited resources. The cost of an intervention may be represented as a function of a particular epidemiological parameter, such as the sufficient contact rate (*Brandeau et al.*, 2003), recovery rate, or rate of movement into a protected state, like vaccination or quarantine (*Nowzari et al.*, 2015). The cost function can also be a function of the number of individuals treated, with that treatment number affecting the epidemiological dynamics (*Goldman and Lightwood*, 2002). The solution methods utilized in epidemiological resource allocation problems are often analytical (see, e.g, *Brandeau et al.*, 2003; *Mbah and Gilligan*, 2011). *Zaric and Brandeau* (2001) use Taylor series approximations to find solutions for a resource allocation model with an underlying SI model and linear production functions.

Yin et al. (2023) consider allocation of ventilators using a multi-stage stochastic model. *Basciftci et al.* (2023) compare stochastic programming and distributionally robust optimization approaches to balance cost and quality of service in COVID-19 test kit and vaccine distribution. Generally, problems related to supply, demand, and matching the two in vaccine distribution amenable to operations research models are outlined by *Dai and Song* (2021). Resource allocation models are also utilized in

pandemic settings to ensure resources are used wisely while satisfying social distancing requirements (see, e.g., *Barnhart et al.*, 2022; *Navabi-Shirazi et al.*, 2022; *Gore et al.*, 2022)

On the other hand, network interdiction models focus on the control of disease spread, and assume that the population has a graphical structure. The epidemiological optimization models are an example of general work that mitigates the spread of contagions by removing nodes (*Albert et al.*, 2000) or edges (*Kimura et al.*, 2009) from a network. *Enns et al.* (2012) formulate the epidemiological edge removal problem as a nonconvex quadratically constrained quadratic program (QCQP) and develop a related QCQP algorithm. *Nandi and Medal* (2016) use mixed-integer linear programs for formulating their epidemiological network interdiction models. *Koch et al.* (2013) consider how the basic reproductive number changes when edges are randomly removed from a contact network.

In addition to basic disease control, a number of studies consider multi-criteria decision making in pandemic conditions, generally examining the interplay between economic and health conditions (see, e.g., *Akbarpour et al.*, 2020; *Silva et al.*, 2020; *Thunström et al.*, 2020; *Birge et al.*, 2020; *Fajgelbaum et al.*, 2020). *Duque et al.* (2020) use a stochastic optimization model with the detailed epidemiological dynamics mentioned in the previous section (*Wang et al.*, 2020) to determine lockdown thresholds, minimizing the total number of days in lockdown while ensuring hospital resources remain available.

3.2.3 Dynamic Control and Uncertain Parameters

One advantage of our model is its ability to consider policy over time, at different phases of an outbreak. The dynamic nature of this decision making is present in a few different papers in epidemiological resource allocation. For instance, *Zaric and Brandeau* (2002) consider a finite-horizon problem, with investment decisions

made over a finite number of time periods, in order to maximize quality-adjusted life years or avoided infections, using heuristic methods for the general problem. *Blount et al.* (1997) use nonlinear programming and dynamic programming approaches to consider control of an SIS epidemic with limited resources. *Du et al.* (2021) use a data-driven optimization approach for multiperiod resource allocation. At each epoch, they estimate parameters with new information, and make an multiperiod decision based on current parameter estimates, though this decision is only used in the next epoch.

Our MDP model for strategic discharges for de-densification is also dynamic. MDPs have a rich theoretical foundation, with a number of methods to optimize long-run discounted cost when the state space and transition probabilities are known. Unfortunately, due to the size of the state space, these parameters are difficult to estimate. In the literature, a few papers formulate epidemiological problems as MDPs. In a paper by *Reluga* (2009), continuous-time MDP theory is used to describe individual strategy in one epidemiological game theoretic setting. The emphasis of the work is on equilibrium behavior, rather than systemic decision making, however. *Bisset et al.* (2009) apply Partially Observable Markov Decision Processes (POMDPs) to infectious outbreaks; on the other hand, their experiments simulate and compare a set of prescribed policies with a main goal of disease control, while we estimate the optimal policy weighing both epidemiological and release costs in this chapter. *Yaesoubi and Cohen* (2011) apply the MDP framework to serialized compartmental models, where individuals flow from disease state to disease state in a single direction (e.g., from Susceptible to Infectious, then from Infectious to Removed).

One application of our multi-criteria MDP framework is de-densification of nursing homes. Control of COVID-19 outbreaks is studied in the nursing home setting by *Smith et al.* (2020), who simulate the distribution of COVID-19 tests using a variety of strategies, to compare their effectiveness and cost. *Barak et al.* (2020) also consider

the testing policy problem, using network-based SEIR simulations and analysis of an exponential infection model. Another application of our MDP is early prison release. *Baycik et al.* (2020) apply MDPs in the criminal justice setting, mitigating city-level drug trafficking. However, the strategic release problem and its inherent competing objectives is not considered for the nursing home, cruise ship, or prison case.

3.3 Strategic Releases and the De-Densification Effect

Central to the formulation of the strategic release problem is the representation of the effects of social distancing. Recall that in this chapter, we assume the facility outbreak is represented by a generic epidemiological model with homogeneous mixing and a susceptible compartment. In this section, we introduce the concept of a *de-densification function*, a way to model the effects of reduced overcrowding, in such a model. We derive necessary conditions for a quadratic function to be a de-densification function. Then, we consider a particular epidemiological model – an SIR DCM with two health-risk levels – and derive sufficient conditions for a quadratic de-densification function to have protective cross-group effects.

Definition III.1. Let $\mathcal{E}$ be an epidemiological model with homogeneous mixing, a susceptible compartment, total population size N, and contact rate parameter $\alpha = \alpha(N)$. The function $\alpha(N)$ is a *de-densification function* if $\alpha(N) \geq 0$ and $\alpha(N)$ increases as population size N increases.

Here, increased population size implies decreased social distancing and in turn, higher contact rate. In our numerical studies, we assume the contact rate is a quadratic function of population size. Taking the first derivative, the next theorem about quadratic functions follows directly from the definition.

Theorem III.2. *Suppose the contact rate parameter is a quadratic function of pop-*

ulation size, $N \in [0, N_{\max}]$, of the form

$$\alpha(N) = \kappa_2 N^2 + \kappa_1 N + \kappa_0. \tag{3.1}$$

If the contact rate is a de-densification function, then the following are true.

- $\kappa_0 \geq 0$.
- *If $\kappa_1 = 0$, then $\kappa_2 \geq 0$. If $\kappa_2 = 0$, then $\kappa_1 \geq 0$. Otherwise, $\kappa_1/2\kappa_2 \geq 0$.*

While the definition ensures that the de-densification function captures the social distancing effect of strategic releases, not all de-densification functions may lead to reduced infection transmission rates. Consider an SIR DCM with health risk levels $\mathcal{H} = \{h_1, h_2\}$, governed by the following ODEs.

$$\frac{dS_h}{dt} = -\beta_h \alpha(N) \frac{\sum_{h' \in \mathcal{H}} I_{h'}}{N} S_h, \ h \in \mathcal{H}, \tag{3.2}$$

$$\frac{dI_h}{dt} = \beta_h \alpha(N) \frac{\sum_{h' \in \mathcal{H}} I_{h'}}{N} S_h - \xi_h I_h, \ h \in \mathcal{H}, \tag{3.3}$$

$$\frac{dR_h}{dt} = \xi_h I_h, \ h \in \mathcal{H}. \tag{3.4}$$

Here, β_h represents the probability a susceptible person with health risk level h becomes infected, given contact with an infectious person, and ξ_h is the recovery rate for an infectious person with health risk level h. The rate of new infections for health risk level h is $-dS_h/dt$. Intuitively, the proportion of susceptible individuals with health level h that become infected is the product of the average number of contacts per person $\alpha(N)$, the probability a contact is infectious $\sum_{h' \in \mathcal{H}} I_{h'}/N$, and the probability of becoming infected given contact with an infectious person, β_h.

While strategic releases reduce the contact rate $\alpha(N)$ and may reduce the number of susceptible individuals with that health risk level, S_h, they also increase the proportion of the population that is infected, $\sum_{h' \in \mathcal{H}} I_{h'}/N$. (Recall that we assume only susceptible individuals are released in this chapter.) Thus, even if contact rate is a

de-densification function, the rate of new infections may not decrease with decreased population size.

We are particularly interested in de-densification functions for which the opposite is true, however. What are outbreaks for which social distancing "slows the spread," in terms of epidemiological model and de-densification function? This is a complex question worthy of further study. We consider a specific model, as well as a particular kind of cross-group spread in the final theorem in this section.

Releasing individuals from a particular health risk level generally has a protective effect for at least some of that subpopulation – the ones that leave the facility. For this reason, we are specifically interested in epidemiological test cases where cross-group protective effects exist. When does releasing individuals from one health-risk level help reduce infection transmission in another health risk level? We consider this question for the SIR DCM with heterogeneous health risk levels, in the following theorem.

Theorem III.3. *Consider an SIR DCM with heterogeneous health risk levels $\mathcal{H}$ and de-densification function $\alpha(N)$, as described in Theorem III.2. Suppose there is at least one infection in the facility and at least one susceptible individual with health risk level $h \in \mathcal{H}$. Assume all releases involve only susceptible individuals in other health risk levels. Then, the rate of new infections for health risk level h increases as population size N increases if and only if*

$$\kappa_2 - \kappa_0/N^2 > 0.$$

Proof. Consider

$$\frac{-dS_h}{dt} = \beta_h \alpha(N) \frac{\sum_{h' \in \mathcal{H}} I_{h'}}{N} S_h.$$

Note that the release action does not change the value of the susceptible population with health risk level h, by assumption. In general, the release action does not involve

any infectious population. This implies that

$$C = \beta_h \sum_{h' \in \mathcal{H}} I_{h'} S_h$$

is a positive constant, regardless of the action taken, and

$$\frac{-dS_h}{dt} = C \frac{\kappa_2 N^2 + \kappa_1 N + \kappa_0}{N}.$$

Taking the derivative with respect to N and noting that $C > 0$ yields the result. □

The following corollaries are immediate.

Corollary III.4. *Under the SIR DCM, if de-densification function α is a linear function of population size N and release actions do not involve health risk level h, then rate of new infections for h does not increase with population size.*

Corollary III.5. *Under the SIR DCM, if de-densification function α is a concave quadratic function of population size N and release actions do not involve health risk level h, then rate of new infections for h does not increase with population size.*

3.4 Decision Making Models

Now that we have considered how epidemiological dynamics are impacted by strategic releases, we can explore how those decisions should be structured. We do so by formulating the epidemiological and demographic state of the system as an MDP. In this section, we give a brief overview of discrete time MDPs, then present the set of parameters that we utilize in our finite-horizon, multi-criteria formulation, followed by the details of the model. While all notation is described within the text, Table 3.1 also provides a quick reference for interested readers.

Category	Symbol	Meaning
MDP	Ω	MDP State Space
	$\mathcal{A}$	MDP Action Space
	$P_{ij}(a)$	Probability of transition from i to j if action a is taken
	$\rho_{\text{ep}}(i,a)$	Expected epidemiological cost of $i \in \Omega$, $a \in \mathcal{A}(i)$
	$\rho_{\text{rel}}(i,a)$	Expected release cost of $i \in \Omega$, $a \in \mathcal{A}(i)$
	λ_{ep}	Weight in [0,1] of epidemiological costs
	$\rho(i,a)$	Weighted cost function
	ℓ	Length of epoch
	T	finite-horizon length
	$\mathcal{H}$	Set of health risk levels
	$\mathcal{R}$	Set of release categories
	$r^* \in \mathcal{R}$	Category ineligible for release
	$\mathcal{V}$	Set of virus states, e.g., S, I, R
	v^i_{hr}	Number of individuals in population with $h \in \mathcal{H}$, $r \in \mathcal{R}$, $v \in \mathcal{V}$
	$W(i)$	Value function for $i \in \Omega$
	$Q(i,a)$	State-action value function for $i \in \Omega$, $a \in \mathcal{A}(i)$
	$\pi(i)$	Optimal action taken in state $i \in \Omega$
SpSa	G	Generative model of the MDP
	d	Current depth of lookahead tree
	H_a	Action horizon length
	H_{final}	Number of epochs no action is taken
	$\lvert C\rvert$	Number of children for each node, action pair
	N_{final}	Number of trajectories to estimate value function of leaves
SpSa-C	$\lvert\overline{C}\rvert$	Number of samples to form representative children
ST-SpSa(-C)	ω	Number of trajectory groups
QLLFA	θ	Weight parameters learned by the algorithm
	k	Iteration of the algorithm
	θ_k	Estimate of θ during iteration k
	$\phi(i,a)$	Basis functions to estimate $Q(i,a)$ for $i \in \Omega$, $a \in \mathcal{A}(i)$
	ϵ	Convergence criterion
Epi Instance	α	Average contact rate per day, whole population
	β_h	Probability of transmission for invidividual with health risk $h \in \mathcal{H}$
	ξ_h	Rate of recovery from infection for invidividual with health risk $h \in \mathcal{H}$
	N	Total facility population
	$N(i,a)$	Population size after release action a occurs in state i
	κ_2, κ_1, κ_0	Constants for quadratic de-densification function (3.1)
Release Action	a_0	none
	a_1	high health, low recidivism only
	a_2	high health only
	a_3	low recidivism only
	a_4	only exclude low health, high recidivism
	a_5	all eligible

Table 3.1: Summary of Notation and Key Parameter

3.4.1 Markov Decision Processes

An MDP can be described by a tuple $(\Omega, \mathcal{A}, P, \rho)$. The set Ω contains the states of the system, while $\mathcal{A}(i)$ contains the actions available for each state $i \in \Omega$, with the set of all actions $\mathcal{A} = \bigcup_{i\in\Omega} \mathcal{A}(i)$. For all states, we assume $\mathcal{A}(i) \supseteq \{a_0\}$, where a_0 is

the null release action. In the discrete time, finite-horizon setting, we have T epochs, each of length ℓ. Every epoch, $t = 0, \ldots, T-1$, a state $i \in \Omega$ is observed, an action $a \in \mathcal{A}(i)$ is taken, and the system transitions to a new state j with probability $P_{ij}(a)$, defined for all $j \in \Omega$. An immediate cost is accrued at this time, given by function $\rho(i, a) : \Omega \times \mathcal{A} \to \mathbb{R}^-$. This represents the expected cost of taking action a while in state i over the course of one epoch.

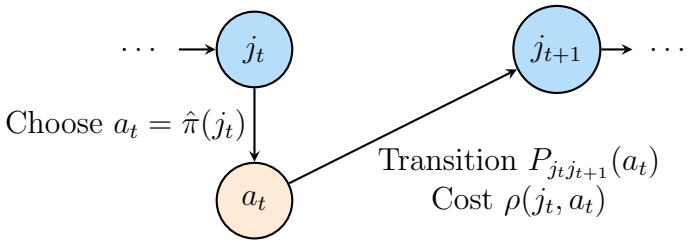


Figure 3.1: Example of an MDP transition during epoch $t = 0, \ldots, T-1$, with current state j_t, action a_t chosen by policy $\hat{\pi}(j_t)$, and transition to next state j_{t+1} governed by probability distribution $P_{j_t \cdot}(a_t)$. Cost $\rho(j_t, a_t)$ is accrued immediately.

The purpose of an MDP is to find a policy function $\pi : \Omega \to \mathcal{A}$ that assigns actions to states in a way that minimizes some cost criterion over time. We consider the expected total discounted cost criterion. Because it is common practice to maximize rewards in the MDP setting, we represent the costs as negative "rewards," and aim to maximize these rewards. In alignment with the application setting, though, we refer to them as costs when describing the meanings of our parameters.

For any policy $\hat{\pi} : \Omega \to \mathcal{A}$, and all states $i \in \Omega$, the expected total discounted cost over time is given by

$$W_{\hat{\pi}}(i) = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t \rho(j_t, \hat{\pi}(j_t)) | j_0 = i\right],$$

where $\gamma \in (0, 1)$ is the discount and j_0 is the initial state at time $t = 0$. The goal is

to find a policy π such that for all $i \in \Omega$,

$$W_\pi(i) = \max_{\hat{\pi}} W_{\hat{\pi}}(i).$$

For this optimal policy π, we denote the value function $W : \Omega \to \mathbb{R}^-$ as $W(i) = W_\pi(i)$ for all $i \in \Omega$. We illustrate an example of an MDP transition in Figure 3.1.

3.4.2 Model Parameters

In our model, the state space describes the phase and demographic composition of a facility outbreak. Each epoch, the facility is divided into virus compartments $\mathcal{V}$ (e.g., $\{S, I, R\}$), that are further divided by demographic characteristics. Within the facility, we assume that there is a heterogeneous population consisting of individuals with discrete health risk level $h \in \mathcal{H}$ and discrete release category $r \in \mathcal{R}$, with $r^* \in \mathcal{R}$ representing the category that is ineligible for release (e.g., due to length of sentence served, or lack of alternatives to the nursing home).

Table 3.2: The table below shows an example state $i \in \Omega$ with virus states $\mathcal{V} = \{S, I, R\}$, health risk levels $\mathcal{H} = \{h_1, h_2\}$, and release categories $\mathcal{R} = \{r_1, r_2, r^*\}$. This is the initial state in our numerical studies.

	S			I	R
	r_1	r_2	r^*	$\mathcal{R}$	$\mathcal{R}$
h_1	**15**	**15**	**29**	**1**	**0**
h_2	**10**	**10**	**20**	**0**	**0**

The states of the system, Ω, are a subset of $\mathbb{Z}^{|\mathcal{V}|\times|\mathcal{H}|\times|\mathcal{R}|}$, with each component of state $i \in \Omega$, denoted v^i_{hr}, equal to the number of individuals that have health risk level h and release category r and are currently in virus state v, for all $h \in \mathcal{H}$, $r \in \mathcal{R}$, and $v \in \mathcal{V}$. An example state, later used as the initial state of the system in our numerical studies, can be viewed in Table 3.2. Notice that some components of the state are aggregated in this representation – with $\sum_{r\in\mathcal{R}} v^i_{hr}$ given for $v \in \{I, R\}$ and

$h \in \mathcal{H}$, while the susceptible population is divided into components by both health risk and release risk level. This is because no releases can occur for infectious or recovered individuals, and release category does not affect epidemiological dynamics. Thus, the value function $W(i)$ is not affected by the release category of infectious or recovered individuals, and the aggregated representation is sufficient.

Release decisions occur at regular intervals of length ℓ. At the beginning of an epoch, the decision maker observes the state of the system, i, and makes a release decision a, in the set of permissible release decisions $\mathcal{A}(i) \subseteq \mathbb{Z}^{|\mathcal{H}|\times(|\mathcal{R}|-1)}$, with each component, denoted a_{hr}, equal to the number of releases in each category $h \in \mathcal{H}$ and $r \in \mathcal{R} \setminus \{r^*\}$. An example action is provided in Table 3.3 and corresponds to the high health, low recidivism action in our numerical studies.

Table 3.3: Example action $a \in \mathcal{A}(i)$ for state $i \in \Omega$, given in Table 3.2. This corresponds to the high health, low recidivism action in our numerical studies.

	r_1	r_2
h_1	**0**	**0**
h_2	**10**	**0**

Expected epidemiological costs, $\rho_{\text{ep}}(i,a)$, and expected release costs, $\rho_{\text{rel}}(i,a)$, for the epoch are accrued. With weight $\lambda_{\text{ep}} \in [0,1]$, we define the overall cost (reward) function as

$$\rho(i,a) = \lambda_{\text{ep}}\rho_{\text{ep}}(i,a) + (1-\lambda_{\text{ep}})\rho_{\text{rel}}(i,a), \quad i \in \Omega,\ a \in \mathcal{A}(i).$$

After cost is incurred, there is a transition from the resulting state after release to next epoch state, j, with probability $P_{ij}(a)$. One possible transition for the state and action given in Tables 3.2–3.3 is given in Table 3.4. This state is taken from a trajectory in our static benchmark simulations.

Table 3.4: Example next state $j \in \Omega$ with virus states $\mathcal{V} = \{S, I, R\}$, health risk levels $\mathcal{H} = \{h_1, h_2\}$, and release categories $\mathcal{R} = \{r_1, r_2, r^*\}$, taken from a static benchmark simulation trajectory.

	S			I	R
	r_1	r_2	r^*	$\mathcal{R}$	$\mathcal{R}$
h_1	**15**	**14**	**28**	**1**	**2**
h_2	**0**	**7**	**16**	**5**	**2**

3.4.3 Generative Markov Decision Processes

If the complete state space and state-action transition probability matrix of an MDP is known, a linear program can be utilized to find the optimal policy (*Ross*, 2014). Because it is difficult to construct the entire state space, which suffers from the curse of dimensionality, in this chapter, we consider solution approaches for an MDP that includes what *Kearns et al.* (2002) call a *generative model*, a black box that simulates transitions from any state-action pair in the space, by randomly sampling the next state and reward from the appropriate distributions, and Q learning with linear function approximation. We describe these approaches in Sections 3.5.1 and 3.5.2.

3.5 Solution Methods

We describe the algorithms that we utilize to solve the multi-objective de-densification problem for pandemic containment. We begin with sparse sampling-based methods and conclude with Q learning with linear function approximation.

3.5.1 Sparse Sampling-Based Methods

The first approach we consider is based on a sparse sampling algorithm developed by *Kearns et al.* (2002). Their algorithm identifies a near-optimal action at the current state, without constructing a full policy table, by constructing a sparse lookahead

tree. By focusing on the current state, this approach provides theoretical guarantees of running time based only on a desired approximation error bound and size of the sampling tree, without respect to the number of states in the MDP.

We first describe the algorithm and discuss the basic adaptations we make for scalability in Section 3.5.1.1. Then, we describe the *Split-Trajectory Sparse Sampling* algorithm we construct for out-of-sample testing in Section 3.5.1.2, followed by the cluster adaptation of both algorithms in Section 3.5.1.3.

3.5.1.1 Baseline Sparse Sampling Algorithm with Adaptations for Scalability

The sparse sampling method constructs a sampling tree with the current state $i \in \Omega$ as the root node. For each action $a \in \mathcal{A}(i)$ available for current state i, the generative model draws a set of samples C from state-action pair (i, a). Each sampled state j becomes a child of the root node, with arc (i, j) labeled with action a and sampled reward ρ_{ij}. This process is repeated for each child node of i, with $|\mathcal{A}(j)||C|$ children of state j at a node are sampled to create new child nodes. This process continues until the tree reaches height H, the horizon size. The value function at the root node is estimated recursively, with the following formulas.

$$\hat{W}_\pi(j) = \begin{cases} 0, \text{ if } j \text{ is a leaf,} \\ \max_{a \in \mathcal{A}(j)} \frac{1}{|C|} \sum_{c \in C(j,a)} (\rho_{jc} + \gamma \hat{W}_\pi(c)), \text{ otherwise.} \end{cases}$$

We incorporate a few changes to the sparse sampling algorithm to reduce computational time. First, we adapt the algorithm to stop branching the sampling tree from any node that represents an absorbing state, reducing the size of the lookahead tree. Second, in our adapted sparse sampling algorithm, we introduce the concept of an action horizon, H_a. Here, actions only can be taken during the first H_a epochs

of horizon T. During the final epochs $H_{\text{final}} = T - H_{\text{a}}$, no actions are taken but the system continues to evolve. This allows us to limit exponential tree growth to horizon H_{a}. Calculating the value function estimate for the leaf nodes then requires sampling a number of trajectories of length H_{final}, rather than simply assuming each action has value 0. Intuitively, the longer the action horizon, the better the algorithm will estimate Q for different initial actions. When $T = H_{\text{a}}$, we recover the original sparse sampling algorithm.

This adaptation applies to generic generative MDPs. In our problem, this assumption allows us to consider the impact of different timing of dynamic decisions early in an epidemic – as granular as week 1 to 2 to 3 – while taking into account the long-term effects of those release decisions in the final epochs of the horizon. Adapting the algorithms developed by *Kearns et al.* (2002) to include these features, we obtain the sparse sampling (SpSa) algorithm, Algorithm 2. This algorithm calls a number of helper functions, each an algorithm itself, labeled Algorithms 3–5.

Algorithm 2 Sparse Sampling (SpSa)

1: **Input:** Action horizon H_{a}, initial state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.
2: **Output:** Action that maximizes estimated Q function at state i.
3: Define $\{(a, \hat{Q}_{H_{\text{a}}}(i,a)) : a \in \mathcal{A}(i)\}$=**EstimateQ**($H_{\text{a}}$, i, γ, G, $|C|$, H_{final}, N_{final})
4: **return** $\arg\max_{a \in \mathcal{A}(i)} \{\hat{Q}_{H_{\text{a}}}(i,a))\}$

3.5.1.2 Split-Trajectory Sparse Sampling Algorithm

The sparse sampling algorithm is an online method, meant to be applied in real time as a trajectory unfolds. The algorithm produces an estimate of the optimal action to take at a particular time, knowing that more actions can be taken in the future as the state of the system evolves. How do policies derived by this method perform over time, in terms of total mean discounted reward? How does this com-

Algorithm 3 EstimateQ

Input: Current depth of lookahead tree d, current state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.
Output: Estimated Q function for current state i and all allowable actions.
if State i is absorbing **then**
  return $\{(a_0, 0)\}$
else if Depth $d = 0$ **then**
  return EstimateFinalW(i, γ, G, H_{final}, N_{final})
else
  for Action $a \in \mathcal{A}(i)$ **do**
    Using generative model G, generate $|C|$ samples of next states C and calculate mean sampled cost $\hat{\rho}(i, a)$
    Estimate $\hat{Q}_d(i, a)$ as follows:

$$\hat{Q}_d(i,a) = \hat{\rho}(i,a) + \gamma \frac{1}{|C|} \sum_{j \in C} \textbf{EstimateW}(d-1, j, \gamma, G, |C|, H_{\text{final}}, N_{\text{final}})$$

  end for
end if
return $\{(a, \hat{Q}_d(i, a)) : a \in \mathcal{A}(i)\}$

Algorithm 4 EstimateFinalW

Input: Current state i, discount γ, MDP generative model G, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.
Output: Estimated value function for state i during final H_{final} epochs.
Using generative model G, generate N_{final} trajectories of H_{final} epochs each from state i, taking null action a_0 each epoch t and noting reward r_{nt}, for $n = 1, \ldots, N_{\text{final}}$.
Calculate mean total discounted reward

$$\hat{W}_0(i, a_0) = \frac{1}{N_{\text{final}}} \sum_{n=1}^{N_{\text{final}}} \sum_{t=0}^{H_{\text{final}}-1} \gamma^t \hat{\rho}_{nt}$$

return $\{(a_0, \hat{W}_0(i, a_0))\}$

pare to objective values obtained by other MDP solution methods? Out-of-sample trajectory simulations can help answer these questions. Following the law of large numbers, the greater the number of trajectories N_{traj}, the better the expected estimation. Unfortunately, applying the algorithm to each trajectory for every epoch

Algorithm 5 EstimateW

1: **Input:** Current depth of lookahead tree d, current state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}.
2: **Output:** Estimated value function for current state i.
3: Define $\{(a, \hat{Q}_d(i,a)) : a \in \mathcal{A}(i)\}$=**EstimateQ**($d$, i, γ, G, $|C|$, H_{final}, N_{final})
4: **return** $\max_{a \in \mathcal{A}(i)}\{\hat{Q}_d(i,a))\}$

requires $N_{\text{traj}} H_a$ calls of the SpSa algorithm, with lookahead trees with horizon $H_a - t$ for $t = 0, 1, \ldots, H_a - 1$. In practical terms, this limits the number of trajectories that can be examined, or forces a decision maker to shorten the action horizon H_a.

To consider the performance of the SpSa algorithm on a large number of new trajectories, we develop the *Split-Trajectory Sparse Sampling* (ST-SpSa) algorithm. In ST-SpSa, at time $t = 0$, all trajectories share the same initial state and the algorithm only needs to be performed once. After the estimated optimal action is taken, the transition for each trajectory can produce up to N_{traj} unique states at $t = 1$. Testing the performance of the online method without adaptation would require performing the sparse sampling algorithm for each unique state. This would be the case for $H_{\text{a}} - 1$ action epochs. Instead, to reduce computational time while considering a large number of trajectories, we perform cluster analysis of the current states at each action epoch after $t = 0$ to identify ω representative states. We add an extra dimension to the current states describing which actions are available at that state to ensure clusters include only states that share an action set. We call these clusters *trajectory groups*, to distinguish them from clusters used in the cluster adaptation. We then perform the sparse sampling algorithm on the ω representative states and apply the action to all states in the respective trajectory group. The full description and algorithmic details can be found in Algorithm 6.

Algorithm 6 Split-Trajectory Sparse Sampling (ST-SpSa)

1: **Input:** Action horizon H_{a}, initial state i, discount γ, MDP generative model G, number of children $|C|$, number of final epochs H_{final}, number of trajectories to estimate value function of leaves N_{final}, number of out-of-sample trajectories N_{traj}, number of trajectory groups ω.
2: **Output:** Set of N_{traj} trajectories of length $H_{\text{a}} + H_{\text{final}}$ with actions taken in first H_{a} epochs.
3: Obtain estimated optimal initial action with **SpSa**(H_{a}, i, γ, G, $|C|$, H_{final}, N_{final}).
4: Generate reward and next states $\hat{\Omega}_1$ from i and estimated optimal action for N_{traj} trajectories.
5: **for** $t = 1, \ldots, H_{\text{a}} - 1$ **do**
6: Identify actions available in each state $j \in \hat{\Omega}_t$. Add dimension with index to separate states with different actions available.
7: Perform cluster analysis on $\hat{\Omega}_t$ to determine ω representative states and corresponding trajectory groups (clusters).
8: Obtain estimated optimal action for each representative state j with **SpSa**($H_{\text{a}} - t$, j, γ, G, $|C|$, H_{final}, N_{final}).
9: Generate reward and next state for each trajectory using the estimated optimal action from its trajectory group. Record set of next states $\hat{\Omega}_t$.
10: **end for**
11: For each trajectory, generate reward and next state for H_{final} final epochs.
12: **return** Trajectories.

3.5.1.3 Cluster Adaptation of Sparse Sampling-Based Methods

While trajectories constructed by the ST-SpSa algorithm demonstrate improvement of at least 7.3% over a static benchmark in 4 of 9 test cases in Section 3.8.2, in five of the test cases, we find the static benchmark performs better. Delving into some of those instances, we notice that the performance of SpSa can be sensitive to the number of absorbing states sampled at the root for a particular action. The children of the root node are random samples, subject to variance. If the number of absorbing states sampled for a particular action at the root is different enough from the expected number of absorbing states, this can impact the action chosen. On the other hand, expanding the number of children near the root of the tree has a multiplicative effect on runtime, even if $|C|$ is only changed at the first level of the tree.

For this reason, we develop an adaptation of SpSa, that utilizes large samples and associated clusters to estimate the children of nodes in the sparse lookahead tree. We demonstrate the effectiveness of this method in our numerical studies, with improvement over SpSa for all 9 instances and improvement over the static benchmark for 8 of 9 instances. We call this algorithm *Sparse Sampling with Clusters* (SpSa-C) and the corresponding out-of-sample test *Split-Trajectory Sparse Sampling with Clusters* (ST-SpSa-C).

SpSa-C works by sampling a large number of next states $\overline{C}$ and corresponding rewards everywhere that SpSa samples smaller number $|C|$ children. The mean immediate reward associated with node i and action a is estimated using set $\overline{C}$. The children of node i, C, are calculated using cluster analysis of the nonabsorbing states in $\overline{C}$. The centers of the $|C|$ clusters become children. Each child c has associated cluster of states $\overline{C}_c$. When calculating the Q function for node i and action a, we use empirical probability associated with each cluster, $\frac{|\overline{C}_c|}{|\overline{C}|}$.

The SpSa-C algorithm steps are the same as those found in Algorithms 2–5 except for a few changes. First, $|\overline{C}|$ becomes an input parameter to any algorithm that requires $|C|$, to satisfy the recursion properties. Second, Algorithm 3, EstimateQ, has the lines given in Algorithm 7, that replace lines 9–10.

Algorithm 7 Cluster Adaptation of EstimateQ – Lines 9–10

1: Using generative model G, generate $|\overline{C}|$ samples of next states $\overline{C}$ and calculate mean sampled cost $\hat{\rho}(i,a)$
2: Perform cluster analysis of nonabsorbing states in $\overline{C}$ to construct $|C|$ children C
3: For $c \in C$, denote the states in its cluster as $\overline{C}_c$
4: Estimate $\hat{Q}_d(i,a)$ as follows:

$$\hat{Q}_d(i,a) = \hat{\rho}(i,a) + \gamma \frac{|\overline{C}_c|}{|\overline{C}|} \mathbf{EstimateW}(d-1, j, \gamma, G, |C|, H_{\text{final}}, N_{\text{final}}, |\overline{C}|)$$

3.5.2 Q Learning with Linear Function Approximation

The second type of approach that we consider is Q learning with linear function approximation, which estimates the state-action value function, or Q function,

$$Q_{\hat{\pi}}(i,a) = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t \rho(i_t, \hat{\pi}(i_t)) | i_0 = i, a_0 = a\right].$$

3.5.2.1 Background

This approach is situated in the field of reinforcement learning. With the large instance size inherent in stochastic epidemiological environments, function approximation (*Baird*, 1995) can be utilized with reinforcement learning, to ensure computationally feasible algorithms. *Libin et al.* (2020) apply deep reinforcement learning, which uses neural networks as the function approximators, to an influenza epidemic control problem. While neural networks are powerful approximators, they also are known to be sensitive to hyperparameter tuning and require environment interaction. The Q learning algorithm with linear function approximation (*Chen et al.*, 2019) is a method that utilizes previously simulated trajectories, and is more easily adaptable to epidemiological settings, allowing historical data or separately generated trajectories to be used to learn the optimal policy. The generative model used in the sparse sampling method can also be applied to create trajectories. The basis functions we develop specific to our test instances can be seen in Section 3.5.2.3.

3.5.2.2 Overview

The Q function represents the total expected discounted reward of taking action a, then following generic policy $\hat{\pi}$. By subtracting the value function from the Q function, we get the advantage of taking action a over following policy $\hat{\pi}$ in state i. The goal of Q learning is to learn the optimal Q function, $Q(i,a) = Q_\pi(i,a)$ for

optimal policy π, which satisfies

$$Q(i,a) = \rho(i,a) + \gamma \sum_j P_{ij}(a') \max_{a'} Q(j,a').$$

If we take a sample trajectory $\{(i_k, a_k)\}_{k=0}^{T-1}$ created by following the optimal policy, then we have the following unbiased estimate:

$$\hat{Q}(i_t, a_t) = \rho(i_t, a_t) + \gamma \max_{a'} Q(i_{t+1}, a').$$

This motivates the iterative Q learning algorithm, with Q estimate Q_k at each iteration k. After randomly initializing the function Q_0, we perform the following update based on a sample trajectory generated with randomized behavior policy $\hat{\pi}$.

$$Q_{k+1}(i,a) = \begin{cases} Q_k(i,a) + \zeta((i,a) + \gamma \max_{a'} Q_k(i_{k+1}, a') - Q_k(i,a)), \text{ if } (i,a) = (i_k, a_k), \\ Q_k(i,a), \text{ otherwise.} \end{cases}$$

Here, we take the difference between the newest estimate of Q^* and the previous estimate, and move toward it with step size ζ for the current state-action pair (and do not update any other state-action pair). The ability to only update one state-action pair at a time means that this tabular method requires a large number of samples.

Linear function approximation addresses this computational challenge by assuming that $Q(i,a)$ is a weighted sum of basis functions, $\phi(i,a)$, with components $m = 1, \ldots, M$ (*Chen et al.*, 2019). With weight vector $\theta \in \mathbb{R}^M$, we have $Q \approx \phi(i,a)^T\theta$. In Q learning with linear function approximation, rather than learning the Q function, we learn the weight parameters θ. One benefit of this method is that the basis functions can incorporate domain knowledge.

We present the algorithmic details of Q learning with linear function approximation in Algorithm 8. We add a convergence criterion that ensures if the distance

between θ_k and θ_{k+1} falls below parameter ϵ times the magnitude of θ_k, the algorithm stops. In the algorithm, again, we have the difference between the current Q estimate and the previous one, and step size ζ. But now, because we are ascending on θ, we multiply by the gradient of Q_θ, which equals $\phi(s_k, a_k)$.

Algorithm 8 Q Learning with Linear Function Approximation for Epidemiological Control.

Input: Total environmental actions K, initial state i_0, randomized behavioral policy $\hat{\pi}$, convergence criterion ϵ, step size ζ, generative model for the MDP G.
Output: Estimate of weight parameter θ.
Initialize $j := 1$, $k := 1$, $i_1 := i_0$. Generate randomized trajectory as follows.
while $j \leq K$ **do**
  if i_k is absorbing state **then**
    Define $a_k := a_0$, the null action. Define $i_{k+1} := i_0$ and $\rho(i_k, a_k) = 0$.
    Increment k by 1.
  else
    Sample action a_k from $\hat{\pi}(i_k, \cdot)$. Sample reward $\rho(i_k, a_k)$ and next state i_{k+1} from G.
    Increment j and k by 1.
  end if
end while
Let $\bar{K}$ denote the total number of state-action pairs in the trajectory.
Initialize parameter $\theta = \theta_0$.
while $k \leq \bar{K}$ and and $\|\theta_{k+1} - \theta_k\| \geq \epsilon \|\theta_k\|$ **do**
  Update θ with gradient ascent as follows (*Chen et al.*, 2019)
  if i_k is not absorbing **then**
    $\theta_{k+1} = \theta_k + \zeta \phi(i_k, a_k) \left(\rho(i_k, a_k) + \gamma \max_a \phi(i_{k+1}, a)^T \theta_k - \phi(i_k, a_k)^T \theta_k\right)$
  else
    Set $\theta_{k+1} = \theta_k$.
  end if
end while
Set $\theta = \theta_k$.

3.5.2.3 Basis Functions

Incorporating domain knowledge into the basis functions in the Q learning method with linear function approximation can improve the algorithm's performance, allowing the functions to better approximate Q. Regardless of epidemiological outbreak or facility type, the Q function for the strategic release problem should include aspects

related to epidemiological cost and release cost. In this section, we give an example of how to develop basis functions for the strategic release problem that incorporate these two cost types. We consider the prison release problem and SIR stochastic ICM with homogeneous mixing that we utilize in our computational tests. First, we describe the epidemiological basis components, and then we give the recidivism-related components.

Suppose the state of the facility at the start of epoch t is i, and we plan to take action a. Because the basis function is meant to help us estimate the Q function for the entire trajectory, not just immediate reward, we construct epidemiological components that estimate the number of new infections in each future epoch $t+\mathrm{w}$ for $\mathrm{w} = 0, \ldots, w-1$, given that action a is taken at time t, and the null action is taken later. Number of epochs to estimate w is a hyperparameter that has to be tuned.

In order to construct these components, we develop a set of recursive functions that estimate the number of new infections for health risk level $h \in \mathcal{H}$ over $w\ell$ days of the epochs, denoted $\tau = 0, \ldots, w\ell - 1$. We define $S^h(\tau)$ and $I^h(\tau)$ to be the number of susceptible and infectious individuals in the facility at the start of day τ and $I^h_{\text{new}}(\tau)$ to be the number of new infections that take place that day. These three functions all depend on state i and action a, though we omit them from the notation for ease of reading.

At the start of the first epoch, the first two functions are defined by the population after release – the number of susceptible and infectious individuals that remain.

$$S^h(0) = \sum_{r \in \mathcal{R}} S^i_{hr} - \sum_{r \in \mathcal{R}_{\text{rel}}} a_{hr} \tag{3.5}$$

$$I^h(0) = I^i_h. \tag{3.6}$$

The number of new infections during day $\tau = 0, \ldots, w\ell - 1$ is estimated using a

formula based on the deterministic compartmental analogue of the stochastic ICM, given in equations (3.2)–(3.4), as follows.

$$I^h_{\text{new}}(\tau) = \min\left\{\beta_h\left(\kappa_2 N(i,a)^2 + \kappa_1 N(i,a) + \kappa_0\right)\frac{\sum_{h'\in\mathcal{H}} I^{h'}(\tau)}{N(i,a)}, 1\right\} S^h(\tau). \tag{3.7}$$

Recall that $\kappa_2 N(i,a)^2 + \kappa_1 N(i,a) + \kappa_0$ is the contact rate for the current facility, while β_h is the probability of transmission for health risk level $h \in \mathcal{H}$. Intuitively, $\beta_h\left(\kappa_2 N(i,a)^2 + \kappa_1 N(i,a) + \kappa_0\right)\frac{\sum_{h'\in\mathcal{H}} I_{h'}}{N(i,a)}$ estimates the expected number of infection transmission events one individual will have in the next time period – the product of their probability of infection given contact, the number of contacts they expect to have, and the probability that a contact will be infected. If this value is less than 1, it is multiplied by the total number of susceptible individuals present in the facility with that health risk level, h, to estimate the total expected number of infectious contacts. Otherwise, we estimate that all susceptible individuals with health risk level h are infected.

To update the number of susceptible individuals for the next day, we subtract those we estimated were infected:

$$S^h(\tau) = S^h(\tau - 1) - I^h_{\text{new}}(\tau - 1). \tag{3.8}$$

To update the number of infectious individuals in the next day, we add those we estimated were infected and then multiply by the probability of not recovering that day.

$$I^h(\tau) = (1 - \xi_h)(I^h(\tau - 1) + I^h_{\text{new}}(\tau - 1)). \tag{3.9}$$

Then, for each epoch, $t + \text{w}$ for $\text{w} = 0, \ldots, w - 1$, and each health risk level $h \in \mathcal{H}$,

we construct the following basis components.

$$\phi_h^{\mathrm{w}}(i,a) = \frac{1}{\sqrt{M}N} \sum_{\tau=\mathrm{w}\ell}^{(\mathrm{w}+1)\ell-1} I_{\mathrm{new}}^h(\tau), \tag{3.10a}$$

$$\phi_h^{\mathrm{w,quad}}(i,a) = \sqrt{M}\left(\phi_h^{\mathrm{w}}(i,a)\right)^2. \tag{3.10b}$$

Our recidivism-related components, linear and quadratic functions of the number released, and bias component are as follows.

$$\phi_r^{\mathrm{lin}}(i,a) = \frac{1}{\sqrt{M}N} \cdot \sum_{h\in\mathcal{H}} a_{hr}, \quad r \in \mathcal{R} \setminus \{r^*\} \tag{3.11a}$$

$$\phi_r^{\mathrm{quad}}(i,a) = \sqrt{M}\left(\phi_r^{\mathrm{lin}}(i,a)\right)^2, \quad r \in \mathcal{R} \setminus \{r^*\} \tag{3.11b}$$

$$\phi_{\mathrm{bias}}(i,a) = \frac{1}{\sqrt{M}}. \tag{3.11c}$$

Recall that M is the number of components of ϕ. Here, the leading coefficients ensure that $||\phi|| \leq 1$.

3.6 Benchmark De-Densification Test Instances

To test our models, we examine the early prison release problem described in Section 3.1, with two health risk levels and three recidivism risk levels (including one ineligible for release), as well as virus states $\mathcal{V} = \{S, I, R\}$.

3.6.1 Epidemiological Outbreak Test Instances

We develop benchmark epidemiological outbreak test instances that can be used in future multi-objective epidemiological MDP work using an extension of the basic stochastic ICM built in to EpiModel. We adapt the existing one-group homogeneous and two-group heterogeneous mixing SIR models to a two-group homogeneous mixing

SIR model with option to track individual non-health features.

3.6.1.1 One-Group SIR Model

In the basic one-group model included in EpiModel, each day, random encounters occur between individuals in the facility, labeled $\{1, \ldots, N\}$, based on the contact rate α (e.g., 5 contacts per person). The total expected number of contact pairs $\bar{z}$ is calculated using the following formula:

$$\bar{z} = \left[\frac{\alpha N}{2} \right].$$

Then, two random samples of size $\bar{z}$ are taken from individuals $\{1, \ldots, N\}$, with replacement, to form vectors $\mathbf{e}^1$ and $\mathbf{e}^2$ of length $\bar{z}$. If any component $i = 1, \ldots, \bar{z}$ of vectors $\mathbf{e}^1$ and $\mathbf{e}^2$ has the property that $e_i^1 = e_i^2$, entry e_i^2 is resampled until that is no longer the case. The ith component of each vector then represents an encounter between individual e_i^1 and e_i^2, for $i = 1, \ldots, \bar{z}$.

For each encounter, the current status of each individual is checked. If one individual is susceptible and another infectious, a random binomial draw determines whether the susceptible individual is infected. Finally, among the infected, random binomial draws determine whether recovery takes place. The random encounter, infection, and recovery steps repeat for each day of the simulation.

This one-group model assumes that probability of transmission and recovery rate are the same for the whole population. This is not ideal for our problem, in which we are interested in balancing various levels of health risk and recidivism risk.

3.6.1.2 Two-Group Heterogeneous Mixing SIR Model

The two-group heterogeneous mixing model built into EpiModel expands upon the one-group model. The basic structure remains the same, with each day composed

of random encounters, infection transmission, and recovery. In contrast to the basic model, however, the extension assumes the facility has two distinct groups, each with their own probability of infection and recovery rate. (In particular, when a susceptible individual in group 1 has contact with an infectious individual in group 2, the random binomial draw uses the group 1 probability of infection.) This property is a better fit for our problem, where we are interested in considering the effect of releasing individuals from a facility with different health and recidivism risk levels.

On the other hand, this built-in extension assumes the groups mix purely heterogeneously, meaning a person from group 1 only interacts with people in group 2, and vice versa. The EpiModel programmers mention this type of model can be appropriate for a sexually transmitted disease under simplifying assumptions about gender and sexual orientation (*EpiModel*, 2022).

For our problem, on the other hand, where we are interested in exploring the effects of de-densification of crowded facilities on epidemiological outbreaks, the homogeneous mixing used in the one-group model is a better fit. For this reason, we adapt the code from the two models to create a new two-group homogeneous mixing model.

3.6.1.3 Two-Group Homogeneous Mixing SIR Model with Non-Health Features

In our adaptation of the stochastic ICM, we simulate random encounters using the one-group model method described in Section 3.6.1.1. The random binomial draws that simulate infection and recovery, however, use group-specific probability and rate, respectively, like the two-group heterogeneous mixing model described in Section 3.6.1.2. This code can be used to simulate two-group homogeneous mixing in general.

In addition, more specific to our problem, the adaptation includes the option to

label each initial susceptible individual with one of three release categories. Then, as these labeled individuals randomly interact and are infected, we keep track of how many susceptible individuals remain in each health and release category. Because release category is no longer relevant for infected and recovered individuals, who are not eligible for release, we only keep track of release category for susceptible individuals. This ensures that we do not consider two states to be different when, in essence, they are the same.

In our code, we refer to release category more generically as a feature. In this way, we distinguish non-health feature from group, which in EpiModel refers to a designation that impacts epidemiological parameters.

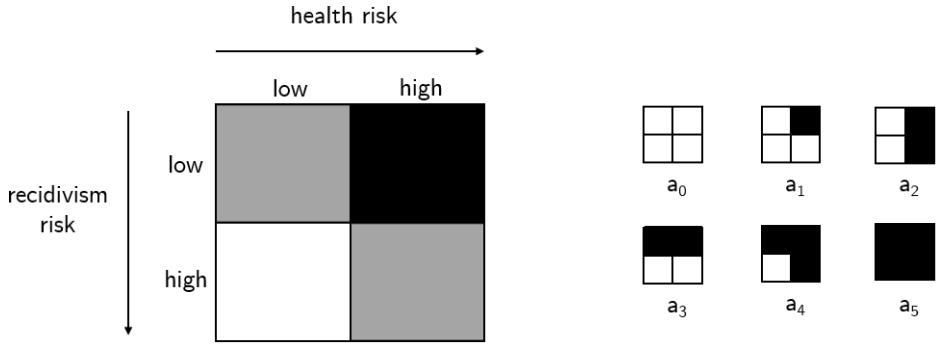
3.6.2 Epidemiological Parameters

Because we have low- and high-risk health levels in our experiments, we assume that the high-risk susceptible population has a higher probability of becoming infected when they come into contact with the pathogen than the low-risk susceptible population. We also assume that the rate of recovery (or the inverse of recovery time) is smaller for the high-risk infected population, as shown in Table 3.5.

Table 3.5: Epidemiological Parameters

	Low health-risk group h_1	High health-risk group h_2
Probability of infection	0.01	0.1
Recovery rate	0.5	0.125

We assume 60% of the 100-person facility is low-health risk level (h_1), and 50% are ineligible for release (r_3). Of those eligible for release, we assume an equal proportion are low recidivism risk (r_1) and high recidivism risk (r_2). We assume that one person is initially infected. Table 3.2 depicts the full initial state of the system. We assume that at full capacity, the average daily contact rate is 10.



(a) Release Priority (b) Action Space

Figure 3.2: (a) Priority for prisoner release. Darker boxes are higher priority. (b) Available actions in each epoch. For each action, all currently susceptible prisoners in the shaded categories are released.

3.6.3 Action Space Definition

In order to test the multi-criteria MDP with these test instances, we need to define an action space for releases. To limit the size of the lookahead tree in the sparse sampling algorithm, we restrict our action space to six possible release actions, illustrated in Figure 3.2. We choose these actions to reflect a prioritization of higher health risk and lower recidivism risk individuals for release, as shown in Figure 3.2a. In each of actions $\{a_0, \ldots, a_5\}$, we release all currently susceptible individuals in particular (h, r) risk categories, as pictured in Figure 3.2b. In action a_0, no one is released; in a_1, only the highest priority individuals are released – those with high health risk and low recidivism risk. In actions a_2 and a_3, we allow either all high health risk or all low recidivism risk populations to be released, which allows us to compare the weight of health versus recidivism risk in the resulting optimal policy. Finally, in a_4, we release all but the lowest priority individuals – those with high recidivism and low health risk, and in a_5, we release all eligible individuals who are currently susceptible.

Note that not all actions are available in every state. In order to release susceptible

Table 3.6: Categories that must have currently susceptible individuals for each action to be taken.

Action	Category
a_0	
a_1	(high health, low rec)
a_2	(high health, high rec)
a_3	(low health, low rec)
a_4	(low health, low rec), (high health, high rec)
a_5	(low health, high rec)

individuals in a certain category, there must be currently susceptible individuals in that category. In Table 3.6, we list the categories that must have currently susceptible individuals in order for an action to be available. For instance, to take action a_4, both (low health, low rec) and (high health, high rec) must have currently susceptible individuals. Otherwise, the appropriate equivalent action to a_4 is either a_2 or a_3. This table ensures that the effect of each action is unique for every state.

Our choice of action space takes advantage of a natural prioritization of individuals for release, and allows us to limit the size of lookahead tree. There are two key characteristics of this action space that help with this. First, there are only six actions, which limits the action-driven growth of the sparse sampling algorithm tree at each epoch to multiplication by six. Second, and more importantly, by releasing *all* susceptible individuals in a particular (h, r) category at state i, we further limit the set of actions available in the next epoch.

To maintain consistency across methods, we use these six actions in our benchmark tests and Q learning with linear function approximation tests as well.

3.6.4 Contact Rate Assumptions

The final epidemiological parameter that we need to define is the *de-densification function*, or how the contact rate changes as the crowding level decreases. We consider three functions with quadratic form (3.1), with coefficients given in Table 3.7. Notice

that at full capacity, the contact rate is 10. All three satisfy the conditions in Theorem III.2 for de-densification functions, and the first two satisfy the conditions given by Theorem III.3, ensuring that release actions affecting one health risk level lower the new infection rate for the other health risk level.

Table 3.7: Quadratic coefficients for de-densification functions.

Contact Rate Assumption	κ_2	κ_1	κ_0
Polynomial	0.00044	0.046	1
Quadratic	0.001	0	0
Proportional	0	0	0.1

3.6.5 Multiobjective Parameters

A key aspect of the strategic release problem is the balancing of competing objectives. We test three reward functions, that either (a) prioritize health, (b) prioritize crime reduction, or (c) give equal priority.

We define health cost $\rho_{\text{ep}}(i, a)$ as the expected number of long-term health consequences from newly infected individuals. Given probability of long-term health effect p_h for $h \in \mathcal{H}$ and number of newly infected individuals I^h_{new}, we have

$$\rho_{\text{ep}}(i, a) = \sum_{h \in H} p_h I^h_{\text{new}}. \tag{3.12}$$

We define release cost $\rho_{\text{rel}}(i, a)$ as the expected number of crimes committed by newly released individuals. Given probability of recidivism p_r for $r \in \mathcal{R} \setminus \{r_3\}$ and number of newly released individuals a^r_{new}, we have

$$\rho_{\text{rel}}(i, a) = \sum_{r=r_1, r_2} p_r a^r_{\text{new}}. \tag{3.13}$$

We assume the probability is larger for high risk than low risk in both cases, as seen in Table 3.8.

Table 3.8: Reward parameters.

	Low risk	High risk
Probability of recidivism (p_{r_1}, p_{r_2})	0.1	0.25
Probability of long-term health effect (p_{h_1}, p_{h_2})	0.1	0.25

For each of the different cost functions, the cost is defined using priority weights λ^{rel} and λ^{health} as

$$\rho(i, a) = -\lambda^{\text{rel}}\rho_{\text{rel}}(i, a) - \lambda^{\text{health}}\rho_{\text{health}}(i, a).$$

The weights we use are shown in Table 3.9. Here, we assume when we prioritize health

Table 3.9: Weight parameters.

Prioritize	γ^{rel}	γ^{health}
Health	1	2
Crime Reduction	2	1
Both Equally	1	1

that one long-term effect is as costly as two crimes, while one crime is as costly as two long-term effects when we prioritize crime reduction.

3.7 Open-Source Generic Markov Simulation Code

To test our methods, we develop a generic Markov simulation framework in R that can be used for other generative MDP problems. This framework includes a number of objects that work together to simulate and analyze discrete Markov chains, including the following.

- **GlobalMarkovParameters.** This class contains attributes that describe the Markov chain, such as num_epochs (length of horizon) and state_components (labels for components associated with a state, e.g., *Infectious,* h_1).

- **TransitionSimulator.** This class simulates the transition from a state, action pair, generating a reward and next state.

- **ActionGenerator.** This class generates the set of actions available at a particular state.

- **ActionSimulator.** This class determines the action to take at a particular state.

- **MarkovTrajSimulator.** This class generates a Markov trajectory of particular length, using GlobalMarkovParameters, TransitionSimulator, and ActionSimulator objects.

- **MarkovResults.** This class analyzes a dataframe of Markov trajectories, calculating useful metrics such as mean total discounted reward for a given set of discounts and weights.

We also implement generic subclasses of ActionSimulator that can be used with discrete Markov chains.

- **ZeroActionSimulator.** This class assigns a zero action, regardless of state. For example, the subclass for our experiments, EpiRelZeroActionSimulator, releases no one from the facility.

- **FixedActionSimulator.** This class takes as input a list of fixed actions to take at particular epochs, regardless of state. For any unspecified epoch, the zero action is taken.

- **RandomActionSimulator.** This class chooses a random action from available actions. The class requires an ActionGenerator object to determine available actions. Options to limit the epochs when actions can be taken or to set a probability of no action taken are available.

- **QActionSimulator.** Given a function $\phi : \Omega \times \mathcal{A} \rightarrow \mathbb{R}^n$ and weights $\theta \in \mathbb{R}^n$, this class determines the action that maximizes $\phi(i, a)^T \theta$ for particular state i.

The benefit of this generic framework, available on Github, is that it allows practitioners to adapt the dynamic methods that we utilize to other problems that can be modeled with MDPs. For instance, we implement Algorithm 6, ST-SpSa, in R, taking these generic classes as input. By creating subclasses specific to new MDPs, the ST-SpSa can be easily utilized for new problems. In addition, MarkovTrajSimulator and RandomActionSimulator can be used to create random trajectories, to be utilized in the Q learning algorithm with linear function approximation, Algorithm 8. When an optimal θ is determined for particular function ϕ, out-of-sample tests can implemented using MarkovTrajSimulator and QActionSimulator.

As an aside, because Algorithm 8 does not require online generation of trajectories via EpiModel, an R program, we implement this Q learning with linear function approximation algorithm in our preferred language, Python. While this code is more tailored to our problem, slight adjustments to the functions that read trajectories and define phi can be made to utilize the algorithm in other contexts.

3.8 Computational Experiments

In this chapter, we introduce three methods, from sparse sampling-based approaches with and without clusters to Q learning with linear function approximation. We also develop nine test instances, taking into account de-densification and reward functions. In this section, we describe the experimental setup and delve into the results, comparing the reward and computational performance associated with each of the three methods across the nine test cases. In general, we find that the cluster method outperforms the baseline SpSa in all cases, as well as Q learning with linear function approximation methods for crime reduction prioritization. The Q learn-

ing method yields similar objective values to ST-SpSa-C for the other two priority weights, with shorter runtimes.

3.8.1 Experimental Setup

For each test instance, we compare the performance of the three different types of policies on a set of 1000 trajectories of 10 epochs each. These trajectories share an initial state, and each has a fixed seed associated with every epoch transition. This ensures that when an action is shared across policies for a specific trajectory, epoch, and state, the transition to the next state is also the same. We consider discount $\gamma = 0.9$.

As a benchmark to evaluate the policies generated by our dynamic methods, we consider static, single-action policies that take place before the first epoch. We consider each of the six actions, running 1000 trajectories after each action is applied, and choose the action with the best mean discounted reward as the benchmark for each test.

We implement ST-SpSa with and without clusters in R, using a generic Markov simulation framework we describe in Section 3.7. For our experiments, we use the following parameters, shown in Table 3.10.

Table 3.10: ST-SpSa parameters.

Parameter	Value
Action horizon H_a	3
Number of trajectories to estimate leaf node Q N_{final}	10
Trajectory groups ω	10
Baseline Children $\lvert C\rvert$	10
Clusters Children $\lvert C\rvert$	5
Clusters Samples $\lvert\overline{C}\rvert$	100

Like the sparse sampling methods, we implement the trajectory sampling portion of Algorithm 8 in R, using the generic Markov simulation framework. The randomized

policy works as follows. At the start of each new trajectory, we choose three random epochs in which an action can be taken. Then, when an action epoch arrives, an action is randomly chosen from available actions. In some cases, only the null action will be available at future action epochs. We use this policy to encourage the agent to "wait and see" during epoch 0 in a high number of trajectories. The number of environmental interactions that we require is 100,000.

We implement the remainder of Q learning with linear function approximation in Python with convergence criteria $\epsilon = 1 \times 10^{-6}$. For each test instance, we consider the set of hyperparameters seen in Table 3.11.

Table 3.11: Q learning with linear function approximation hyperparameters tested.

Initial θ Seed	**Step Type**	**Step Size**	w
0	div_root	0.5	1
1	div_linear	0.75	2
		1	3

The step ζ is a function of algorithm iteration k and step size. When the step type is div_root and step size is .75, $\zeta = 0.75/\sqrt{k}$; when step type is div_linear, $\zeta = 0.75/k$. The initial weight θ_0 is randomly initialized to values in $[-1, 0]$ using random seeds 0 or 1.

Because reinforcement learning methods are known to be sensitive to hyperparameters, we tune them for each of the 9 test instances separately, and choose the resulting θ that gives the best out-of-sample value.

3.8.2 Comparison Across Methods

Table 3.12 shows how the out-of-sample objective changes when the policies given by each method are followed, rather than the best single release action. Notice that the sparse sampling algorithm with clusters and Q learning method lead to the highest percentage improvement in the health prioritization cases, as well as similar perfor-

Table 3.12: Percentage improvement in total mean discounted reward for different methods over static benchmark, for different contact assumptions and priority weights.

	Polynomial			Quadratic			Proportional		
Method	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**
ST-SpSa	-22.9	-10.3	9.0	-27.0	-10.5	11.6	9.7	-2.2	7.3
ST-SpSa-C	13.3	1.3	10.0	5.0	-0.4	13.0	11.4	0.2	8.7
QLLFA	13.3	0.9	-0.9	5.7	0.5	2.5	11.6	0	-0.6

mance to the static benchmark in the equal priority cases. When crime reduction is highest priority, ST-SpSa-C performs best. In Section 3.8.3, we delve into the polynomial case and demonstrate that the dynamic ST-SpSa-C method chooses a largely static policy for equal priority weight, explaining its lack of improvement.

Without the cluster adaptation, ST-SpSa yields varying results – with improvement as high as 11.6% and decrement as high as 27%. Incorporating k-means clustering to the algorithm improves consistency. In Table 3.13, we explore why this may be the case. Here, we note the probability of the epidemic dying out in the first epoch after a particular release action is taken, over the 1000 out-of-sample benchmark trajectories. Because of the large number of trajectories, we expect that this empirical probability is our best estimate of the true probability. We then examine the sparse

Table 3.13: Probability of epidemic dying out after one epoch for each action in static benchmark trajectories and sparse sampling lookahead trees, with and without clusters.

	Polynomial			Quadratic		
Action	**Benchmark**	**ST-SpSa-C**	**ST-SpSa**	**Benchmark**	**ST-SpSa-C**	**ST-SpSa**
a_0	0.616	0.590	0.600	0.616	0.590	0.600
a_1	0.722	0.710	0.600	0.702	0.780	0.400
a_2	0.801	0.780	0.900	0.839	0.870	0.900
a_3	0.729	0.730	0.900	0.764	0.750	0.700
a_4	0.806	0.820	0.800	0.857	0.870	1
a_5	0.814	0.840	0.700	0.890	0.880	0.900

lookahead trees for each of the two methods, comparing the sampled child nodes at the initial state root. For the quadratic contact rate assumption and high health,

low recidivism action a_1, for example, 702 out-of-sample trajectories had 0 infections after 1 epoch. For the baseline ST-SpSa sparse lookahead tree, after action a_1 was taken, 4 of 10 sampled child nodes had 0 infections, yielding a probability of .400 of the epidemic dying out. With the cluster adaptation, on the other hand, 78 of 100 sampled children had 0 infections, yielding a much better estimation of the empirical probability of the epidemic dying out, of .780. Here, we see that using a larger sample size with k-means clustering to generate the child nodes better estimates this probability. In turn, the clustering method more consistently outperforms a static benchmark in out-of-sample mean total discounted cost.

Table 3.14: Runtime (hours) for each computational method.

	Polynomial			**Quadratic**			**Proportional**		
Method	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**	**Health**	**Equal**	**Crime**
ST-SpSa	7.04	15.89	16.52	6.76	12.58	12.92	10.6	11.98	13.19
ST-SpSa-C	9.42	9.31	11.63	9.16	9.35	10.99	10.03	10.08	11.87
QLLFA	8.39	8.48	8.35	7.82	7.70	7.68	5.99	6.08	6.03

At the same time, the computational runtime of ST-SpSa-C is generally similar or better than ST-SpSa, as seen in Table 3.14. Only in the cases where ST-SpSa strongly underperforms in terms of objective – the polynomial and quadratic prioritize health instances – is the runtime slightly shorter. Thus, adding the cluster adaptation generally improves objective performance while maintaining runtime. The runtime for Q learning with linear function approximation is shorter than both other methods, though the objective performance is more variable for this method across priority weights.

3.8.3 Comparison Across Policy Weights

To analyze how the policy changes for different priority objectives, we consider the polynomial contact rate assumption. Because the cluster method outperforms the baseline sparse simulator algorithm and shows more consistent performance over

Q learning with linear function approximation across priority weights, we consider the policies given by this method.

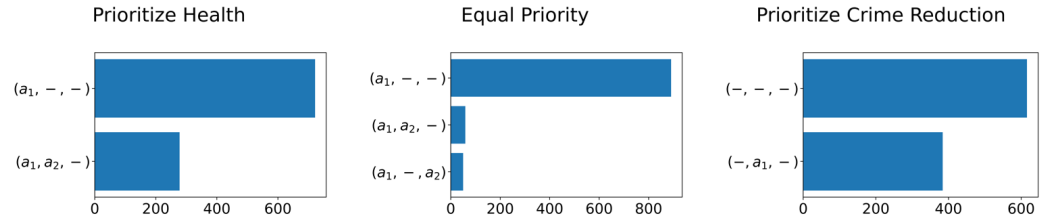


Figure 3.3: Number of 1000 out-of-sample trajectory policies given by cluster method of each type, for polynomial contact rate assumption and three priority weights.

We illustrate the distribution of the 1000 out-of-sample trajectory policies for each priority weight in Figure 3.3. Notice that the instances with greater improvement over the static benchmark, prioritize health and crime reduction, benefit most from the dynamic nature of the cluster method, with at least 20% of trajectories utilizing a release action after one or two epochs. In line with intuition, the initial action is most aggressive for the health prioritization and equal priority cases, a_1, where all individuals with high health and low recidivism risk are released, as opposed to no initial action taken for crime reduction prioritization.

Another point of interest in this figure is that releasing all low recidivism risk individuals (a_3) is never a chosen action. When more individuals than the high health and low recidivism risk group are released, the policies release all high health risk (a_2).

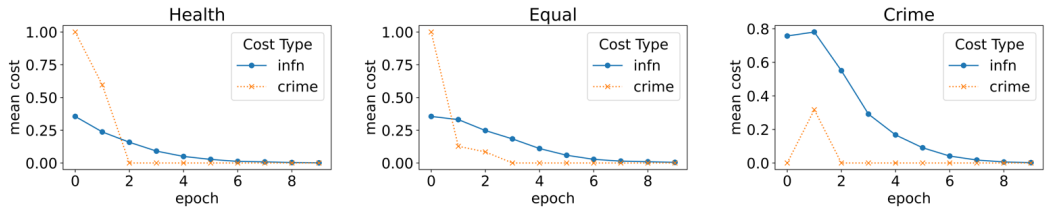


Figure 3.4: Mean cost by cost type over planning horizon for different priority weights under polynomial contact rate assumption for out-of-sample tests using cluster algorithm policy.

Next, we examine the costs associated with the chosen policies for each prioritization type. Figure 3.4 shows the mean infection and crime cost each epoch for out-of-sample tests of the cluster algorithm policy for the polynomial contact rate assumption, under different priority weights. Notice that long-term health effects exceed crime throughout the planning horizon when we prioritize crime reduction. For the equal priority weight, crime exceeds long-term health effects for one epoch, and when we prioritize health, this is the case for two epochs.

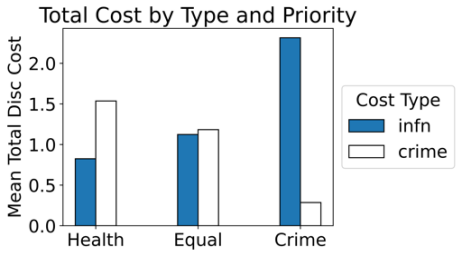


Figure 3.5: Total mean discounted cost by cost type for different priority weights under polynomial contact rate assumption for cluster algorithm out-of-sample tests.

In Figure 3.5, we illustrate the mean total discounted cost for each priority weight for the entire planning horizon. The total discounted crimes committed exceed total discounted long-term health effects for the health prioritization and equal priority objectives. This trend is reversed for the crime reduction prioritization. Here, the priority weights are able to produce distinct policies with the desired differentiated effects.

3.9 Concluding Remarks

In this chapter, we considered the novel de-densification for pandemic containment problem, developing models and solution methods to implement strategic releases that weigh epidemiological and release-related risk. We introduced the problem, noted its key features, and constructed a multi-criteria, multidimensional MDP to model facility population, decisions, and associated costs. We modeled the social

distancing effect of reduced crowding with the concept of a de-densification function, and derived theoretical properties under particular assumptions. We also developed general approaches with open-source software implementations for epidemiological MDPs.

As part of our numerical studies, we also constructed freely available benchmark epidemiological test instances, that can be used in future research. We developed basis functions for Q learning with linear function approximation specific to these test instances and demonstrate their effectiveness in health prioritization cases.

In order to evaluate online sampling-based methods with a large number of independent trajectories, we developed the *Split-Trajectory Sparse Sampling* algorithm. We used this algorithm to compare the performance of online sampling-based methods, with and without cluster adaptations, with Q learning with linear function approximation and a static benchmark. We found that adding cluster adaptations to the online sampling-based methods yielded improved performance over the static benchmark under two priority weights and a number of contact rate assumptions.

This area of research is new, and provides many opportunities for innovation. One area that could benefit from future study is the de-densification function. The basis functions could also be adapted to try to better approximate the Q function in crime reduction prioritization cases.

CHAPTER IV

Stochastic Network Interdiction-Based Mitigation of Fake News

4.1 Introductory Remarks

Peer-to-peer networks have become a major source of news in the contemporary age (*Manjoo*, 2017). "Fake news," purposefully fabricated stories meant to provoke readers, can spread quickly from user to user (*Allcott and Gentzkow*, 2017) and platforms like Facebook have developed strategies for their detection and mitigation (*Iosifidis and Nicoli*, 2020). While mitigation strategies may vary in effectiveness, as discussed in Chapter I, most fake news mitigation models assume that systemic actions are deterministically effective. In this chapter, we propose a new approach to interrupt the flow of misinformation through a network that considers uncertainty in intervention effect: minimizing expected maximum flow with stochastic network interdiction.

Stochastic network interdiction is a variant of classical network interdiction, a type of Stackelberg game (*von Stackelberg and Peacock*, 1952) that involves a leader and a follower. In network interdiction, the leader alters arcs in a graph in some manner, such as reducing capacity or extending length, in order to mitigate the follower's objective, under a budget constraint. One example of network interdiction

is maximizing the shortest path through a network (*Israeli and Wood*, 2002). Another is interdiction of maximum flow (*Wollmer*, 1964). An overview of network interdiction and its variants is given by *Smith and Song* (2020).

In the classical network interdiction setting, the leader and follower know all problem data – including the structure of the network and the effect of arc interdiction. As we consider the flow of fake news through a social media network, however, we have a few forms of uncertainty. If a user sees a piece of information from a malicious source, will they believe it and share it? If the social media network intervenes and attaches a warning to a post, will it prevent the user from sharing it? With extensive data about user behavior on social media available, user actions can be modeled by probability distributions. This characteristic of the fake news mitigation problem makes it amenable to stochastic network interdiction (*Cormican et al.*, 1998), in particular, where some problem data is unknown, but can be described with stochastic models.

In this chapter, we consider a social media network with vulnerable, malicious, and general users. Our goal is to minimize the expected number of vulnerable users who receive information from malicious users, by removing malicious accounts or intervening on user-user links, with actions like adapting newsfeed algorithms or adding flags to posts. We assume these interventions have a certain user-specific probability of preventing an individual from sharing the information. The differentiation of vulnerable users from general users allows the problem to model the spread of hate speech, defining vulnerable users as the targeted protected group. The problem also generalizes to mitigate the flow of fake news to all users in a network if all non-malicious nodes are labeled vulnerable target nodes.

To model this problem, we define and construct a specific supergraph from a given social media network. An example can be seen in Figure 4.1. The structure of this graph allows us to consider both node and edge removal interventions, as well as formulate the number of vulnerable users reached as a maximum flow problem.

We apply the stochastic network interdiction methods developed by *Janjarassuk and Linderoth* (2008) to this graph to minimize the expected number of vulnerable users reached.

Contributions of the chapter. To the best of our knowledge, we are the first to utilize the minimizing expected maximum flow version of stochastic network interdiction to model fake news mitigation. This allows us to incorporate potential ineffectiveness of systemic actions, as opposed to other works described in Chapter I, that assume system-initiated interdiction is deterministic. We adapt the model for scalability, taking advantage of community structure that exists in many social networks. We also develop theoretical results about the nature of fake news mitigation with stochastic intervention on single-source paths and use these results to develop an ϵ-optimal algorithm for our problem on these networks. We demonstrate our theoretical results with empirical examples, using computational implementations of our integer program and algorithm, and illustrate the use of our community structure model on a social network with 23,505 nodes, based on the IMDb actors dataset.

4.2 Mathematical Model

In this section, we give an overview of the graph that we construct to represent the number of vulnerable users who receive fake news. Next, we give our model that formulates the number of target nodes reached as a maximum flow problem. Finally, we represent the fake news mitigation problem using stochastic network interdiction.

4.2.1 Social Media Network Parameters

In our models, we assume that we have a social media network with users U, divided into malicious source nodes, $S = \{s_i : i = 1, \ldots, |S|\}$, target nodes, $T = \{t_i : i = 1, \ldots, |T|\}$, and intermediary nodes, $R = \{r_i : i = 1, \ldots, |R|\}$. The set S represents a group of individuals who each independently share a piece of fake

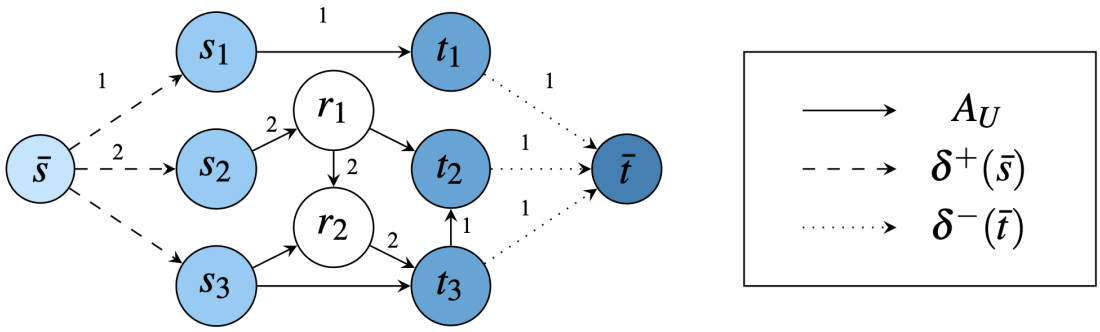


Figure 4.1: Graph $G = (N, A)$ with flow variable y_ω edge labels for scenario $\omega \in \Omega$. Edges with no labels have 0 flow.

news with their followers, while T represents their target audience. This can be a specific group, such as particular targets of hate speech or minor children, or the entire set $U \setminus S$. The set $R = U \setminus (S \cup T)$, which may be empty, represents users who are neither targets nor sources of misinformation, but who may inadvertently spread misinformation through the network. User nodes are connected in the graph by arcs A_U. If user j follows user i, then arc $e = (i, j) \in A_U$. Together, these nodes and arcs constitute a graph $G_U = (U, A_U)$.

The objective of our models is to minimize the expected number of target nodes that receive the fake news, by interdicting particular arcs or source nodes in the social media network. To do so, we formulate the number of target nodes reached as a maximum flow problem on social media network supergraph $G = (N, A)$, where $N = U \cup \{\bar{s}, \bar{t}\}$, with single source node $\bar{s}$ and target node $\bar{t}$, and $A = A_U \cup \delta^+(\bar{s}) \cup \delta^-(\bar{t})$. Here, we define arcs $\delta^+(\bar{s}) = \{(\bar{s}, s) : s \in S\}$ from $\bar{s}$ to each user source node in S and $\delta^-(\bar{t}) = \{(t, \bar{t}) : t \in T\}$ from each user target node to $\bar{t}$. An example can be seen in Figure 4.1.

In our maximum flow model, arcs in $\delta^-(\bar{t})$ each have capacity 1, while all other arcs have infinite capacity. When we maximize flow through G, the bottlenecks are the inarcs of target node $\bar{t}$. The reason we define capacity this way is to make an

equivalency between total maximum flow and number of target nodes reached. Due to flow balance constraints, positive flow on arc $(t, \bar{t}) \in \delta^-(\bar{t})$ implies that the flow of fake news reaches target node t. By constraining flow variables to integer values, we have that a particular target node is reached if and only if the flow from that node to $\bar{t}$ is 1. The predecessors of $\bar{t}$ in G are precisely T, by definition. Thus, the maximum flow problem gives the number of target nodes reached by the flow of fake news.

We define two types of interventions that the social media network can take to interrupt the flow of fake news through the network. We call these actions *system-initiated interdiction.* First, the network can remove a certain number of malicious users. Because this is generally a fully effective action taken by the network, we assume that type of intervention is deterministic. Second, the network can attempt to "remove" a certain number of user-user links. The removal of $e = (i, j) \in A_U$ represents an intervention that prevents j from either seeing the news from i, believing the news from i, or wanting to share the news from i. We assume the success of this would-be intervention on arc $e = (i, j)$ is a binary random variable $\tilde{\xi}_{ij}$ with probability p_e^{ξ} of success.

We also assume, independent of system-initiated interdiction, that a user may choose not to believe another user themselves. We call this *user-initiated interdiction.* For each arc $e = (i, j) \in A_U$, we have binary variable $\tilde{\alpha}_{ij}$. We assume this variable is 1 with probability p_e^{α}, representing user j choosing not to believe news from user i, independent of system-initiated interdiction.

Note that the random variables involved in our model are $\tilde{\xi} \in \{0, 1\}^{|S|+|A_U|}$ and $\tilde{\alpha} \in \{0, 1\}^{|A_U|}$. We can enumerate the support of these random variables as a finite set of scenarios Ω. Each scenario $\omega \in \Omega$ has an associated realization ξ_ω and α_ω, and probability p_ω.

Formally, for each scenario $\omega \in \Omega$ and edge $(i, j) \in A$, we define flow variable $y_{ij\omega} \in \mathbb{Z}_+$. For target node $t \in T$, $y_{t\bar{t}\omega} \geq 1$ if misinformation reaches target node t.

For user arcs $(i, j) \in A_U$, $y_{ij\omega} \geq 1$ if i shares the misinformation and j receives it, on a directed path to a target node. Finally, for $s \in S$, $y_{\bar{s}s\omega} \geq 1$ if s shares the fake news with their followers, on a directed path to a target node. Otherwise, $y_{ij\omega} = 0$. Take Figure 4.1 as an example. The news flows from source node s_2 to r_1, followed by r_2, to t_3, then finally t_2. It is possible when r_1 shares the news that t_2 may see it as well, but because the model only cares whether t_2 is reached or not, not how many times, the flow on the arc (r_1, t_2) is 0. The fact that multiple paths exist to t_2 matters, however, as we start to interdict arcs.

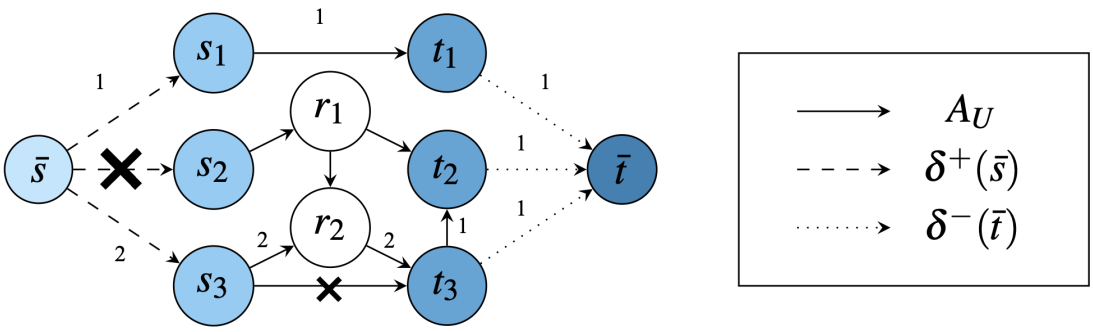


Figure 4.2: Graph $G = (N, A)$ with interdiction. System-initiated interdiction is represented by a large X, and user-initiated interdiction by a small X. The graph includes flow variable y_ω edge labels for scenario $\omega \in \Omega$. Edges with no labels have 0 flow.

For edge $(i, j) \in A_U \cup \delta^+(\bar{s})$, we define system-initiated interdiction variable $x_{ij} \in \{0, 1\}$. For arcs in A_U, this interdiction represents a user-user link intervention. For user source node $s \in S$, $x_{\bar{s}s} = 1$ if the social media network suspends the user. In Figure 4.2, for example, the social media network shuts down malicious user s_2 and target user t_3 is suspicious of malicious user s_3, but willing to believe fake news from intermediary user r_2.

4.2.2 Stochastic Network Interdiction Formulation

Denote the number of target nodes that receive fake news for given system-initiated interdiction decision x and realization ω as $f(x, \omega)$. The stochastic network interdiction model, adapted from *Janjarassuk and Linderoth* (2008), is as follows.

$$\min_{x} \sum_{\omega} p_\omega f(x, \omega) \tag{4.1}$$

$$\text{s.t.} \sum_{(i,j)\in A_U} x_{ij} \leq B_U, \tag{4.2}$$

$$\sum_{(i,j)\in \delta^+(\bar{s})} x_{ij} \leq B_S, \tag{4.3}$$

$$x_{ij} \in \{0, 1\},\ (i, j) \in A \setminus \delta^-(\bar{t}). \tag{4.4}$$

Here, we wish to choose the system-initiated interdiction within arc-interdiction and node-interdiction budgets, B_U and B_S, respectively, that minimizes the expected number of target nodes reached.

The number of target nodes reached for a particular system-initiated interdiction decision x and uncertainty realization ω is found as follows.

$$f(x, \omega) = \max_{y,v} v_\omega \tag{4.5}$$

$$\text{s.t.} \sum_{j\in\delta^-(i)} y_{ji\omega} - \sum_{j\in\delta^+(i)} y_{ij\omega} = \begin{cases} -v_\omega,\ i = \bar{s} \\ 0,\ i \in U \\ v_\omega,\ i = \bar{t}, \end{cases} \tag{4.6}$$

$$y_{ij\omega} \leq \begin{cases} |T|(1 - x_{ij}),\ (i, j) \in \delta^+(\bar{s}) \\ |T|(1 - x_{ij}\xi_{ij\omega})(1 - \alpha_{ij\omega}),\ (i, j) \in A_U \\ 1,\ (i, j) \in \delta^-(\bar{t}), \end{cases} \tag{4.7}$$

$$y_{ij\omega} \in \mathbb{Z}^+,\ (i, j) \in A, \tag{4.8}$$

$$v_\omega \in \mathbb{Z}^+. \tag{4.9}$$

Here, constraints (4.6) act as flow balance constraints. Constraints (4.7) ensure the flow on any successfully system- or user-interdicted arcs is 0. These constraints also ensure that the flow from each target node to the sink node is 0 or 1. Note that these bottlenecks ensure that flow is no more than $|T|$ on any particular arc. Thus, our infinite capacity arcs technically have an upper bound $|T|$ on flow.

If the target to sink node flow $y_{t\bar{t}\omega}$ is 1 for a particular solution, the flow balance constraints imply the target node t is reached in that solution. If the target to sink node flow $y_{t\bar{t}\omega}$ is 0 and t can be reached, there is a path from the source node to t with infinite capacity arcs (and flow at most $|T|-1$), and we can add 1 to that path flow, increasing the objective value, a contradiction. Thus, we have that the objective gives the number of reachable target nodes.

In our model, we assume the flow variables are integer; however, Theorem 6.5 in *Ahuja et al.* (1993) proves that any maximum flow problem with integer arc capacities has an integer optimal flow. For this reason, we can omit the integrality constraints in the optimization problem (4.5) to find $f(x,\omega)$, instead requiring $y_{ij\omega} \geq 0$ for $(i,j) \in A$ and $v_\omega \geq 0$. Then, following *Janjarassuk and Linderoth* (2008), we take the dual of (4.5), yielding the following. Note that we define $\xi_{\bar{s}s} = 1$ and $\alpha_{\bar{s}s\omega} = 0$ for $s \in S$ to unify notation for arcs A_U and $\delta^+(\bar{s})$.

$$f(x,\omega) = \min_{w,h} \sum_{(i,j)\in\delta^-(\bar{t})} h_{ij\omega} + \sum_{(i,j)\in A_U \cup \delta^+(\bar{s})} |T|(1 - x_{ij}\xi_{ij\omega})(1-\alpha_{ij\omega})h_{ij\omega} \tag{4.10}$$

$$\text{s.t. } w_{s\omega} - w_{t\omega} \geq 1, \tag{4.11}$$

$$h_{ij\omega} - w_{i\omega} + w_{j\omega} \geq 0,\ (i,j) \in A, \tag{4.12}$$

$$h_{ij\omega} \geq 0,\ (i,j) \in A. \tag{4.13}$$

Putting together models (4.1) and (4.10), we have

$$\min_{x} \sum_{\omega} p_\omega \left(\sum_{(i,j)\in\delta^-(\bar{t})} h_{ij\omega} + \sum_{(i,j)\in A_U\cup\delta^+(\bar{s})} |T|(1-x_{ij}\xi_{ij\omega})(1-\alpha_{ij\omega})h_{ij\omega} \right) \tag{4.14}$$

$$\text{s.t.} \sum_{(i,j)\in A_U} x_{ij} \le B_U, \tag{4.15}$$

$$\sum_{(i,j)\in\delta^+(s)} x_{ij} \le B_S, \tag{4.16}$$

$$w_{s\omega} - w_{t\omega} \ge 1,\ \omega \in \Omega, \tag{4.17}$$

$$h_{ij\omega} - w_{i\omega} + w_{j\omega} \ge 0,\ (i,j) \in A, \omega \in \Omega, \tag{4.18}$$

$$h_{ij\omega} \ge 0,\ (i,j) \in A, \omega \in \Omega, \tag{4.19}$$

$$x_{ij} \in \{0,1\},\ (i,j) \in A \setminus \delta^-(\bar{t}). \tag{4.20}$$

Notice the second summation in the objective includes terms of the form $x_{ij}h_{ij\omega}$, the product of two variables. *Janjarassuk and Linderoth* (2008) show that (4.14) is equivalent to the following linear program.

$$\min_{x} \sum_{\omega} p_\omega \left(\sum_{(i,j)\in\delta^-(\bar{t})} h_{ij\omega} + \sum_{(i,j)\in A_U\cup\delta^+(\bar{s})} |T|(h_{ij\omega} - z_{ij}\xi_{ij\omega})(1-\alpha_{ij\omega}) \right) \tag{FN}$$

$$\text{s.t.} \sum_{(i,j)\in A_U} x_{ij} \le B_U, \tag{4.21}$$

$$\sum_{(i,j)\in\delta^+(s)} x_{ij} \le B_S, \tag{4.22}$$

$$z_{ij\omega} \le x_{ij},\ (i,j) \in A \setminus \delta^-(\bar{t}), \omega \in \Omega, \tag{4.23}$$

$$z_{ij\omega} \le h_{ij\omega},\ (i,j) \in A \setminus \delta^-(\bar{t}), \omega \in \Omega, \tag{4.24}$$

$$w_{s\omega} - w_{t\omega} \ge 1,\ \omega \in \Omega, \tag{4.25}$$

$$h_{ij\omega} - w_{i\omega} + w_{j\omega} \ge 0,\ (i,j) \in A, \omega \in \Omega, \tag{4.26}$$

$$h_{ij\omega} \ge 0,\ (i,j) \in A, \omega \in \Omega, \tag{4.27}$$

$$z_{ij\omega} \ge 0,\ (i,j) \in A \setminus \delta^-(\bar{t}), \omega \in \Omega, \tag{4.28}$$

$$x_{ij} \in \{0,1\},\ (i,j) \in A \setminus \delta^-(\bar{t}). \tag{4.29}$$

We implement this program in Python with Gurobi with empirical instances in Section 4.4.

4.2.3 Adaptation for Scalability

Implementation of our combined minimization model can be computationally expensive for large social media networks. In this section, we give an alternative formulation of the stochastic network interdiction model for a community-based transformation of a social media network.

Suppose that the user nodes U of a large social media network G_U are partitioned into k communities, $C_1, \ldots, C_k$, using a cluster analysis or hierarchical clustering method. We define the community graph of the social network $G_U^c = (U^c, A^c)$ as follows. The set of nodes $U^c = \{1, \ldots, k\}$ each represent a community. For $i \neq j \in U^c$, there is an arc $(i,j) \in A^c$ if and only if an arc exists from a user in community C_i to a user in community C_j in A_U. We label the set of all arcs from users in community C_i to users in community C_j as E_{ij}. Note that E_{ij} is a subset of A_U.

We assume that once a densely connected community becomes "infected" with misinformation, the misinformation spreads quickly throughout that group, with no user-initiated interdiction. Because of the speed of infection in the community, we assume system-initiated interdiction does not take place within communities either. The emphasis of this scalable model is mitigation of fake news spread between communities.

Within the community graph, each community node is labeled as a source, target, or general node, analagously to the social media network. Source community nodes, labeled S^c, are communities that contain malicious source nodes. Target community nodes, T^c, are communities that contain target nodes and no source nodes. We label

the set of target nodes contained in community C_i as T_i, for each community target node $i \in T^c$. General community nodes R^c are communities that contain only general nodes. The community supergraph $G^c = (N^c, A^c)$ is defined analagously to the social network supergraph $G = (N, A)$, with $N^c = U^c \cup \{\bar{s}, \bar{t}\}$ and $A^c = A^c_U \cup \delta^+(\bar{s}) \cup \delta^-(\bar{t})$. The one change we make when we define G^c is the capacity of the inarcs of the dummy target node. Rather than making each capacity 1, the capacity of arc $(i, \bar{t}) \in A^c$ becomes the number of vulnerable target nodes in community C_i, $|T_i|$. Because other arcs in the supergraph remain infinite capacity, if community C_i that contains target nodes is reachable in the community graph, maximum flow will saturate the capacity of the arc $(i, \bar{t})$. Thus, maximizing flow over this supergraph is equivalent to determining the number of target nodes outside of communities with source nodes that receive fake news. Because we assume misinformation spreads within a community immediately, we can add the number of target nodes in communities with source nodes to this objective to get the total number of target nodes reached.

When the system performs interdictions on nodes or edges in the community supergraph, the corresponding action on the original social media supergraph is much larger. System-initiated interdiction of community source node i implies interdiction of $|C_i|$ nodes in the community. Similarly, system-initiated interdiction of edge (i, j) in the community supergraph corresponds to interdiction of $|E_{ij}|$ edges in the original social media supergraph. We adapt the budget constraints on system-initiation interdiction to reflect this. Overall, the minimization problem is adapted as follows.

$$\min_x \sum_\omega p_\omega \left(\sum_{(i,j)\in\delta^-(\bar{t})} |T_i| h_{ij\omega} + \sum_{(i,j)\in A^c_U \cup \delta^+(\bar{s})} |T| (h_{ij\omega} - z_{ij}\xi_{ij\omega})(1 - \alpha_{ij\omega}) \right) \tag{FN-Comm}$$

$$\text{s.t.} \sum_{(i,j)\in A^c_U} |E_{ij}| x_{ij} \leq B_U, \tag{4.30}$$

$$\sum_{(i,j)\in\delta^+(\bar{s})} |C_j| x_{ij} \leq B_S, \tag{4.31}$$

$$z_{ij\omega} \leq x_{ij},\ (i,j) \in A^c \setminus \delta^-(\bar{t}), \omega \in \Omega, \tag{4.32}$$

$$z_{ij\omega} \leq h_{ij\omega},\ (i,j) \in A^c \setminus \delta^-(\bar{t}), \omega \in \Omega, \tag{4.33}$$

$$w_{\bar{s}\omega} - w_{\bar{t}\omega} \geq 1,\ \omega \in \Omega, \tag{4.34}$$

$$h_{ij\omega} - w_{i\omega} + w_{j\omega} \geq 0,\ (i,j) \in A^c, \omega \in \Omega, \tag{4.35}$$

$$h_{ij\omega} \geq 0,\ (i,j) \in A^c, \omega \in \Omega, \tag{4.36}$$

$$z_{ij\omega} \geq 0,\ (i,j) \in A^c \setminus \delta^-(\bar{t}), \omega \in \Omega, \tag{4.37}$$

$$x_{ij} \in \{0,1\},\ (i,j) \in A^c \setminus \delta^-(\bar{t}). \tag{4.38}$$

As the size of the community graph grows, the number of calculations required to enumerate the stochastic distribution of scenarios grows exponentially. In our computational studies, for instance, the community graph has 189 nodes. While this is much reduced from the original social network, with 23,505 nodes, enumerating all scenarios is computationally infeasible.

For this reason, we use sample average approximation, estimating the stochastic distribution with Monte Carlo sampling. We solve (FN-Comm) M times, each with an independent Monte Carlo sample Ω_m of size n, yielding candidate solution $\hat{x}_m$. Then, using large independent Monte Carlo sample $\bar{\Omega}_1$ of size $N > n$, we solve the maximum flow problem for each scenario $\omega \in \bar{\Omega}_1$, estimating the objective value for each candidate solution $\hat{x}_m$. We then choose the candidate solution with lowest objective value, $\bar{x}$, and perform one final test, with a second independent sample, $\bar{\Omega}_2$ of size N, to estimate the mean number of target nodes reached.

4.3 Path Networks with Stochastic System Interdiction

In fake news mitigation, intuition suggests to interdict arcs as close to source nodes as possible – curbing the spread of misinformation before it gains momentum. When does it make sense to delay system-initiated interdiction? To examine this problem more closely, we look at social media networks that can be represented as paths, $\mathcal{P}_n = (U, A_U)$, $n \in \mathbb{N}$, with one malicious source node, s_1, and n target nodes, $\{t_1, \ldots, t_n\}$ and user arcs $A_U = \{(s_1, t_1)\} \cup \{(t_i, t_{i+1}) : i = 1, \ldots, n-1\}$, as depicted in Figure 4.3. Denote each arc $(\cdot, t_i) \in A_U$ as e_i. For conciseness, denote $p^{\alpha}_{e_i}$ as p^{α}_i and $p^{\xi}_{e_i}$ as p^{ξ}_i. Denote the probabilities of failure to interdict as q^{α}_i and q^{ξ}_i, respectively. This network type, while simple, is a subgraph of most real social media networks. Its simplicity helps us gain insight into when delaying system-initiated interdiction makes sense.

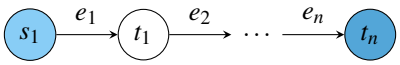


Figure 4.3: Graph $\mathcal{P}_n$.

In this section, we assume the budget allows precisely one arc $e_i \in A_U$ to be interdicted by the system. When system-initiated interdiction is deterministic, regardless of user-initiated interdiction scenario, earlier interdiction is always better. The first arc in the path whose cost is within budget should be chosen, to minimize the expected number of target nodes reached. If system-initiated interdiction is stochastic, on the other hand, when does it make sense to delay interdiction?

Our integer program (FN) gives us one way to determine which arc $e_i \in A_U$ minimizes expected target nodes reached. Another approach involves calculating the expected value, given a particular edge is interdicted by the system. We can repeat this process for all edges in A_U or some subset, if we reduce the size of the search space.

We develop solution methods that utilize this approach in this section. We start by deriving the explicit form of the expected value for this problem in the following theorem.

Theorem IV.1. *Suppose a social media network is described by graph $\mathcal{P}_n$ in Figure 4.3, and $j \in \{1, \ldots, n\}$. The expected number of target nodes reached when edge e_j is interdicted by the system, denoted $\bar{v}(x_{e_j})$, is given by*

$$\sum_{k=1}^{n} kP(\tilde{v} = k | x_{e_j} = 1), \tag{4.39}$$

where for $k \in \{1, \ldots, n\}$,

$$P(\tilde{v} = k | x_{e_j} = 1) = p_{k+1} \prod_{i=1}^{k} q_i. \tag{4.40}$$

Here, the probability of failure to interdict arc $i = 1, \ldots, n$ is given by

$$q_i = \begin{cases} q_i^{\alpha}, \ i \neq j, \\ q_i^{\alpha} q_i^{\xi}, \ i = j, \end{cases} \tag{4.41}$$

and the probability of successful interdiction of arc $i = 1, \ldots, n$ is given by

$$p_i = \begin{cases} p_i^{\alpha}, \ i \neq j, \\ p_i^{\alpha} + p_i^{\xi} - p_i^{\alpha} p_i^{\xi}, i = j, \end{cases} \tag{4.42}$$

while $p_{n+1} = 1$.

Proof. The number of target nodes reached is $k \in \{1, \ldots, n\}$ when edges $e_1, \ldots, e_k$ are not interdicted by system or user, and edge e_{k+1} is. When $\tilde{v} = n$, interdiction along all arcs has failed. These facts are reflected in equation (4.40) and $p_{n+1} = 1$. For $i \neq j$, the success or failure to interdict comes just from the user, which leads

to probabilities p_i^α and q_i^α, respectively. On the other hand, when $i = j$, failure to interdict implies both system and user failed, while success implies either system or user failed. The probabilities in (4.41)–(4.42) reflect this. □

4.3.1 Two Target Nodes

Suppose the number of target nodes $n = 2$. In this section, we show that delayed interdiction is optimal when the probability of the second arc's success, p_2^ξ, exceeds a particular threshold, which is the product of the first arc's probability of success, p_1^ξ, and a threshold multiplier, based on the second arc's user-initiated interdiction probability, p_2^α. An illustration of the threshold multiplier, as a function of p_2^α, can be seen in Figure 4.4. Notice as the probability of successful user interdiction in the second arc increases, the required size of successful system interdiction probability in the second arc in relation to the first arc grows very quickly. In many cases, early interdiction is preferred over later interdiction!

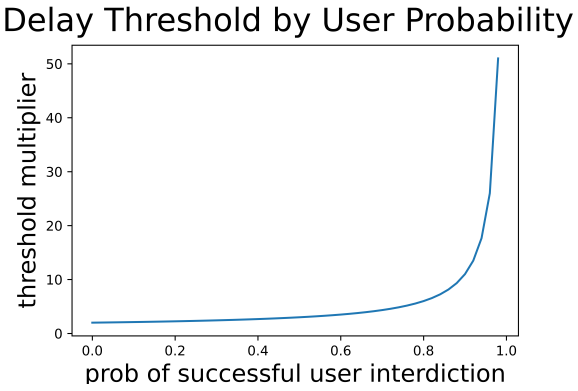


Figure 4.4: The threshold multiplier in the delayed system-initiated interdiction threshold, $1 + 1/q_2^\alpha$, as a function of user-initiated interdiction probability p_2^α.

Theorem IV.2. *Consider graph $\mathcal{P}_n$ in Figure 4.3 with $n = 2$. Suppose the budget allows system-initiated interdiction of either e_1 or e_2. Assume user-initiated interdiction is not deterministic (i.e., $p_i^\alpha < 1$ for $i = 1, 2$).*

Arc e_2 minimizes the expected number of target nodes reached if and only if

$$p_2^\xi \geq (1 + 1/q_2^\alpha)p_1^\xi. \tag{4.43}$$

Proof. We prove the result by showing that (4.43) holds if and only if $\bar{v}(x_{e_1}) - \bar{v}(x_{e_2}) \geq 0$. First, we use Theorem IV.1 to calculate $\bar{v}(x_{e_1}) - \bar{v}(x_{e_2})$. Note that we have the following probability distributions.

$$P(\tilde{v} = k | x_{e_1} = 1) = \begin{cases} p_2^\alpha q_1^\alpha q_1^\xi, \ k = 1, \\ q_1^\alpha q_1^\xi q_2^\alpha, \ k = 2. \end{cases} \tag{4.44}$$

$$P(\tilde{v} = k | x_{e_2} = 1) = \begin{cases} (p_2^\alpha + p_2^\xi - p_2^\alpha p_2^\xi) q_1^\alpha, \ k = 1, \\ q_1^\alpha q_2^\alpha q_2^\xi, \ k = 2. \end{cases} \tag{4.45}$$

Because q_1^α is a positive common factor in all terms of $\bar{v}(x_{e_1}) - \bar{v}(x_{e_2})$, the expression is nonnegative if and only if $\bar{v}(x_{e_1}) - \bar{v}(x_{e_2})/q_1^\alpha \geq 0$. Thus, $\bar{v}(x_{e_1}) - \bar{v}(x_{e_2}) \geq 0$ if and only if

$$(p_2^\alpha q_1^\xi - p_2^\alpha + p_2^\xi - p_2^\alpha p_2^\xi) + 2(q_1^\xi q_2^\alpha - q_2^\alpha q_2^\xi) \geq 0 \tag{4.46}$$

$$\iff (p_2^\alpha - p_2^\alpha p_1^\xi - p_2^\alpha + p_2^\xi - p_2^\alpha p_2^\xi) + 2q_2^\alpha(p_2^\xi - p_1^\xi) \geq 0 \tag{4.47}$$

$$\iff p_2^\alpha(p_2^\xi - p_1^\xi) + p_2^\xi + 2q_2^\alpha(p_2^\xi - p_1^\xi) \geq 0 \tag{4.48}$$

$$\iff (1 + q_2^\alpha)(p_2^\xi - p_1^\xi) + p_2^\xi \geq 0 \tag{4.49}$$

$$\iff -p_1^\xi + q_2^\alpha p_2^\xi - q p_1^\xi \geq 0 \tag{4.50}$$

$$\iff p_2^\xi \geq (1 + 1/q_2^\alpha)p_1^\xi. \tag{4.51}$$

Note that the last line holds because $q_2^\alpha = 1 - p_2^\alpha > 0$. □

Theorem IV.2 has a few immediate corollaries.

Corollary IV.3. *For graph $\mathcal{P}_2$, if $p_1^\xi = p_2^\xi$, then interdiction of arc e_1 is preferred over interdiction of arc e_2, regardless of user-initiated interdiction probabilities.*

Corollary IV.4. *For graph $\mathcal{P}_2$, the preferred interdiction decision depends on system-initiated interdiction and e_2 user-initiated interdiction probabilities. The decision is independent of the e_1 user-initiated interdiction probability.*

4.3.2 ϵ-Optimal Approximation Algorithm for Path Networks

When we consider whether to delay system-initiated interdiction along a path $\mathcal{P}_n$, we compare the expected number of target nodes reached if an edge is interdicted to earlier edges in the path. Independent of system-initiated interdiction, we have user-initiated interdiction. Intuitively, the further and further we move along the path, delaying system-initiated interdiction, the more likely we are to reach a point where user-initiated interdiction plays the greater role in expected number of target nodes reached. In this section, we show that theoretical proof supports this intuition. The distance between values in the sequence $\{\bar{v}(x_{e_i})\}_{i=1}^n$ decreases as the index increases. We describe this phenomenon in more detail in the theorem below, and follow with an algorithm that uses this fact to produce ϵ-optimal solutions to the minimum expected target nodes reached problem for paths of the form $\mathcal{P}_n$.

Theorem IV.5. *Consider graph $\mathcal{P}_n$ in Figure 4.3. For this graph, expected number of nodes reached*

$$|\bar{v}(x_{e_i}) - \bar{v}(x_{e_{i+1}})| \leq \prod_{j=1}^{i-1} q_{e_j}^{\alpha} \sum_{j=i-1}^{n} j \tag{4.52}$$

for $i = 2, \ldots, n-1$. The sequence

$$\left\{ \prod_{j=1}^{k} q_{e_j}^{\alpha} \sum_{j=k}^{n} j \right\}_{k=1}^{n-2} \tag{4.53}$$

is monotone decreasing.

Proof. First, note that for $k = 1, \ldots, i-2$, $P(\tilde{v} = k | x_{e_i} = 1) = P(\tilde{v} = k | x_{e_{i+1}} = 1) = p_{k+1}^{\alpha} \prod_{j=1}^{k} q_j^{\alpha}$, since no system-initiated interdiction is involved. This implies

$$|\bar{v}(x_{e_i}) - \bar{v}(x_{e_{i+1}})| = \left| \sum_{k=i-1}^{n} k(P(\tilde{v} = k | x_{e_i} = 1) - P(\tilde{v} = k | x_{e_{i+1}} = 1)) \right| \tag{4.54}$$

$$\leq \sum_{k=i-1}^{n} k \left| P(\tilde{v} = k | x_{e_i} = 1) - P(\tilde{v} = k | x_{e_{i+1}} = 1) \right|. \tag{4.55}$$

Now, note two cases for the value $P(\tilde{v} = k | x_{e_j} = 1)$. If $j < k+1$, we have that system-initiated interdiction failed, and $P(\tilde{v} = k | x_{e_j} = 1) = p_{k+1} \prod_{l=1}^{k} q_l^{\alpha} q_j^{\xi}$. Otherwise, $P(\tilde{v} = k | x_{e_j} = 1) = p_{k+1} \prod_{l=1}^{k} q_l^{\alpha}$. In either case, for $k \geq i-1$, we have

$$\frac{P(\tilde{v} = k | x_{e_j} = 1)}{\prod_{l=1}^{i-1} q_l^{\alpha}} \in (0, 1), \tag{4.56}$$

which implies

$$\prod_{j=1}^{i-1} q_{e_j}^{\alpha} \frac{\left| P(\tilde{v} = k | x_{e_i} = 1) - P(\tilde{v} = k | x_{e_{i+1}} = 1) \right|}{\prod_{j=1}^{i-1} q_{e_j}^{\alpha}} \leq \prod_{j=1}^{i-1} q_{e_j}^{\alpha}. \tag{4.57}$$

This, along with (4.55), yields the result. □

We utilize this theorem to develop the following ϵ-optimal algorithm. Note that Algorithm 9 checks if the RHS of (4.52) is less than the tolerance ϵ divided by an upper bound on the remaining number of edges, $n - k$. Because sequence (4.53) is monotone decreasing, when this line is satisfied for some k, the difference between $\bar{v}(x_{e_{k+1}})$ and $\bar{v}(x_{e_j})$ is less than ϵ for all $j \geq k+2$.

Algorithm 9 PathInterdict

Input: Optimal value tolerance ϵ. Path graph $\mathcal{P}_n$.
Output: ϵ-optimal interdiction solution.
Initialize $q := q^{\alpha}_{e_1}$ and $C := n(n+1)/2$.
for $k = 1, \ldots, n-2$ **do**
  if $qC < \epsilon/(n-k)$ **then**
    break
  end if
  Update $q := q * q^{\alpha}_{k+1}$ and $C := C - k$.
end for
Find expected number of target nodes reached $\bar{v}(x_{e_j})$ for $j = 1, \ldots, k+1$.
return $\arg\min_{j=1,\ldots,k+1} \bar{v}(x_{e_j})$.

4.4 Numerical Studies

4.4.1 Path Networks with Two Target Nodes

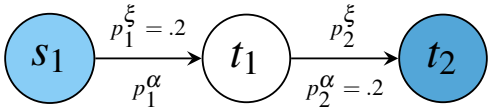


Figure 4.5: Path network $\mathcal{P}_2$ used in our numerical studies.

In Section 4.3.1, we prove that for a path network $\mathcal{P}_n$ with $n = 2$ target nodes, with budget to interdict one user arc, system-initiated interdiction should be delayed from e_1 to e_2 if and only if

$$p^{\xi}_2 \geq (1 + 1/q^{\alpha}_2)p^{\xi}_1.$$

We test this result empirically with our integer program (FN) and $p^{\alpha}_2 = p^{\xi}_1 = .2$, as seen in Figure 4.5. The value p^{ξ}_2 that makes this expression hold with equality is .45. For this reason, we test $p^{\xi}_2 = .44$ and .46. To test whether the arc chosen is independent of p^{α}_1, as Corollary IV.4 implies, we also test two values: p^{α}_1=.1 and .9.

Because the four variables $\tilde{\alpha}_{e_1}$, $\tilde{\alpha}_{e_2}$, $\tilde{\xi}_{e_1}$, and $\tilde{\xi}_{e_2}$ are binary, there are 16 scenarios

$\omega \in \Omega$, each with p_ω calculated by multiplying the four independent probabilities together. For instance, the scenario where all four succeed has $p_\omega = p_1^\alpha p_2^\alpha p_1^\xi p_2^\xi$, while the scenario where all four fail has $p_\omega = q_1^\alpha q_2^\alpha q_1^\xi q_2^\xi$. The results can be seen in Table 4.1.

Table 4.1: Interdicted edge and expected number of target nodes reached for $\mathcal{P}_2$ test cases.

p_2^ξ	p_1^α	$e_j : x_{e_j} = 1$	$\bar{v}(x_{e_j})$
0.44	0.1	e_1	1.2960
0.44	0.9	e_1	0.1440
0.46	0.1	e_2	1.2888
0.46	0.9	e_2	0.1432

Notice when $p_2^\xi = 0.44 < (1 + 1/q_2^\alpha)p_1^\xi$, there is no delay to system-initiated interdiction. On the other hand, when $p_2^\xi = .46$ is greater than the threshold, system-initiated interdiction is delayed until edge e_2. As expected, this delay is independent of initial self-interdiction probability p_1^α, though this probability does affect the ultimate expected number of target nodes reached. The stochastic network interdiction model performed these four tests in less than a second, including scenario generation. The expected values found by the integer program are the same as those found by applying Theorem IV.1 and calculating expected values directly.

4.4.2 ϵ-Optimal Algorithm for Path Networks

In Section 4.3.2, we develop an approximate algorithm that finds an ϵ-optimal solution to the $\mathcal{P}_n$ interdiction problem by reducing the size of the search space, then finding the expected number of target nodes reached if each arc in the search space is chosen. We test the efficacy of this method by comparing the runtime with a naive benchmark. We find the expected number of target nodes reached for every arc in the path, and compare this to the algorithm. We consider a path with $n = 100$ target nodes, and three test cases. In each, $p_i^\xi = .5$, for all arcs e_i, $i = 1, \ldots, 100$.

Self-interdiction probability is also consistent, with a test case for constant p_i^α=.1, .5, and .9, for all arcs e_i, $i = 1, \ldots, 100$. The results can be seen in Table 4.2.

Table 4.2: Search space size and computational runtime for $\mathcal{P}_{100}$ test cases for ϵ-optimal algorithm, compared to naive benchmark.

		Runtime (s)	
p_i^α	Search Space	Benchmark	Alg 9
0.1	100	2.515	2.453
0.5	39	2.462	0.973
0.9	13	2.458	0.327

Notice that the algorithm reduces search space and runtime most for higher probabilities of user-initiated interdiction.

4.4.3 Scalable Mixed-Integer Program for Networks with Community Structure

In Section 4.2.3, we develop an adaptation of the stochastic network interdiction model that takes advantage of the community structure that exists in many social networks and discuss the use of Sample Average Approximation to further reduce computational time.

To test these solution methods, we construct a social network with 23,505 nodes using a subset of the actors born 1990 or later in the IMDb dataset downloaded February 28, 2023. In the graph, an edge exists between two actors if they have worked on a project together. We use the largest connected component, containing about 90% of original nodes, to construct the social network.

To construct the community graph, we use Clauset-Newman-Moore greedy modularity maximization (*Clauset et al.*, 2004), implemented by NetworkX (*Hagberg et al.*, 2008), partitioning the nodes into 189 communities.

We consider system-initiated interdiction of edges in A_U^c, excluding source node interdiction from our analysis. We test different values for the following parameters:

number of community source nodes, budget, and type of random behavior. We consider cases with 1, 2, or 3 community source nodes, selecting the largest communities. We test budgets of 500, 1000, and 1500 for system-initiated edge interdiction.

Table 4.3: Random Behavior Parameters

	p^α	p^ξ
Stochastic User	0.5	1
Stochastic System	0	0.5
Both Stochastic	0.5	0.5

Within type of random behavior, we consider three cases, summarized in Table 4.3. In the first, stochastic users, we have user self-interdiction with probability p_e^α=.5 for all $e \in A_U^c$. For this case, system-initiated interdiction is deterministic and effective, with p_e^ξ=1 for all $e \in A_U^c$. In the second case, stochastic system, no users self-interdict, meaning p_e^α=0 for all $e \in A_U^c$. In this case, system-initiated interdiction is stochastic, and effective with probability p_e^ξ=.5 for all $e \in A_U^c$. Finally, in the last case, both stochastic, users self-interdict with p_e^α=.5 for all $e \in A_U^c$ and the system is effective with probability p_e^ξ=.5 for all $e \in A_U^c$. Intuitively, for fixed budget and number of community source nodes, the expected number of reachable nodes should grow as we move from test cases with stochastic users to both stochastic to stochastic system.

To estimate the optimal solution of the community structure model (FN-Comm), we perform SAA with $m = 5$ Monte Carlo samples with 100 scenarios each to yield 5 candidate solutions. We then estimate the objective value of each candidate solution with an independent Monte Carlo sample with 1000 scenarios, solving the maximum flow problem for each scenario and taking the average. We choose the candidate solution with minimum expected number of target nodes reached. Finally, with the chosen candidate solution, we perform these second-phase SAA tests again with a second independent Monte Carlo sample, also with 1000 scenarios, solving the maxi-

mum flow problem for each scenario and taking the average, to estimate the objective value of the chosen candidate solution. For all 27 test instances, we find the SAA optimality gap is between -.15 and 2.54%. The final estimated objective value is given in Table 4.4.

Table 4.4: Final Objective Value by Test Instance

$\|S^c\|$	Budget	Stochastic	Final Obj Val
1	500	System	14912.6
		Both	13784.8
		User	8905.6
	1000	System	14774.6
		Both	13465.2
		User	5737.8
	1500	System	14717.4
		Both	13346.2
		User	3704.8
2	500	System	12449.6
		Both	11321.8
		User	6442.6
	1000	System	12312.2
		Both	11000.4
		User	3981.9
	1500	System	12254.6
		Both	10883.3
		User	2209.4
3	500	System	10209.9
		Both	9324.6
		User	6021
	1000	System	10072.6
		Both	9143.9
		User	3754.2
	1500	System	10015.4
		Both	9033.4
		User	2233.8

Notice that the expected number of target nodes reached is highest when system-initiated interdiction is stochastic and user-initiated interdiction is deterministically zero, and lowest when the system is deterministically effective and users are stochas-

tically participating in self-interdiction, for fixed budgets and community source node sizes. For fixed community source node sizes, increasing budget has greatest effect on the stochastic users case, where system-initiated interdiction is deterministically effective.

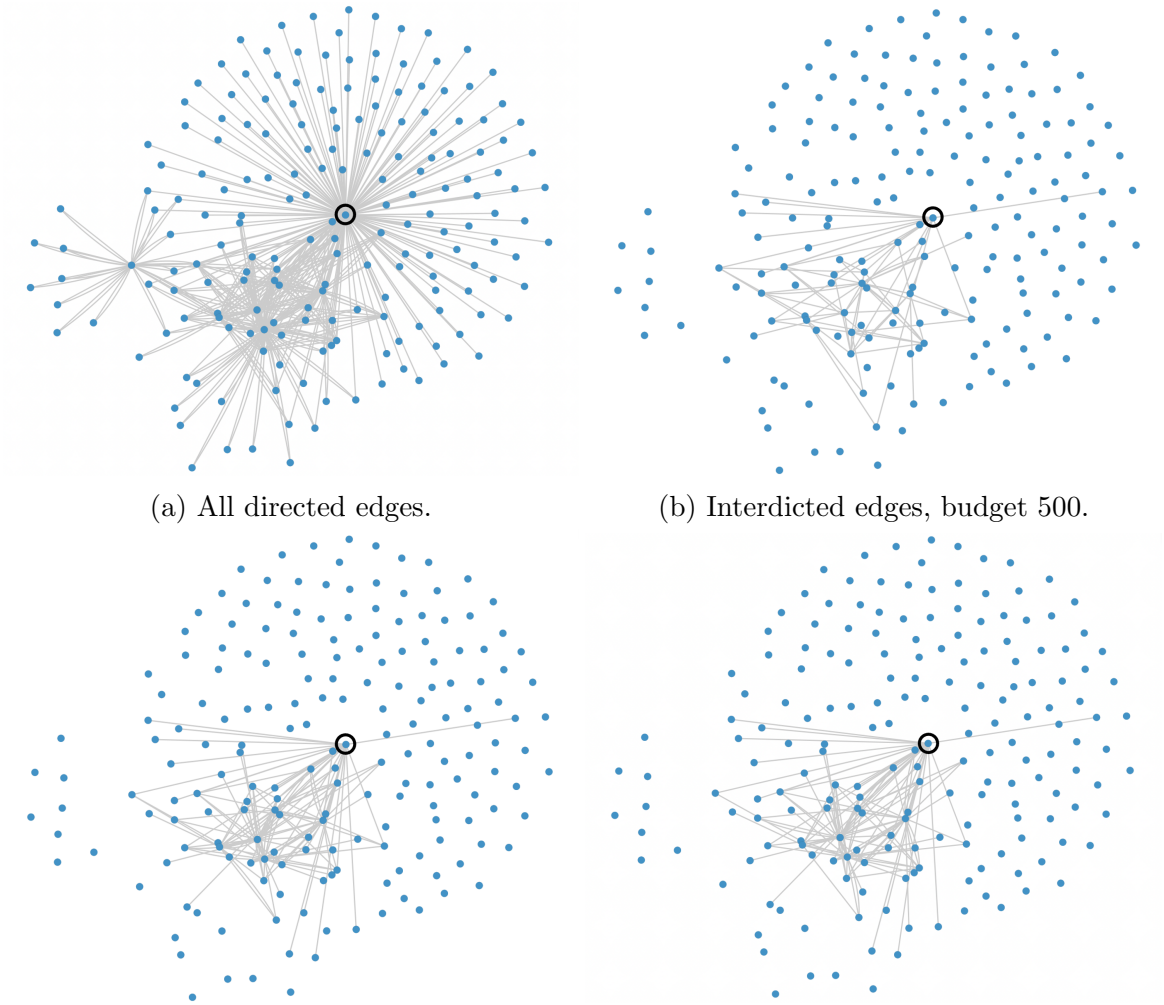
What edges does the mixed-integer program choose to interdict? What are the properties of these edges? We delve into this question for the cases when $|S^c| = 1$, illustrated in Figures 4.6–4.8. In each figure, part (a) shows the community supergraph G^c with directed edges. We exclude dummy nodes $\bar{s}$ and $\bar{t}$ to simplify the visualization. The community source node is circled; all other nodes are community target nodes. Then, in parts (b)–(d), we show the edges that are interdicted by the MIP, for budgets 500, 1000, and 1500, respectively.

Notice that when only users are stochastic, and system-initiated interdiction is deterministically effective, as seen in Figure 4.6, the interdicted edges are clustered in a densely connected area. When only the system is stochastic, and users deterministically do not self-interdict, as seen in Figure 4.8, the interdicted edges are spread throughout the graph. When both are stochastic, as seen in Figure 4.7, we see a pattern between these two extremes, with some interdicted edges in the densely connected area and some spread throughout the graph.

In addition to differences in the community supergraph structure of solutions for the three types of random instances, we see distinctions in original social media network properties of the interdicted edges as well. Recall that each node in U^c represents a community of nodes in U, whereas each arc (i, j) in A^c_U represents a set of edges $E_{ij} \subset A_U$. In Table 4.5, we examine the size of community nodes involved in interdicted edges for each test instance. To do so, we create a vector with two components for each interdicted arc, (i, j), with values $|C_i|$ and $|C_j|$. We exclude any components associated with S^c. Then, we take the median. Notice that a similar pattern exists for each fixed value $|S^c|$ and budget. When only the

Table 4.5: Size of Communities in Interdicted Edges by Test Instance

$\|S^c\|$	Budget	Stochastic	Median Nodes
1	500	System	6
		Both	229
		User	690
	1000	System	6
		Both	140
		User	779
	1500	System	11
		Both	97
		User	490
2	500	System	5
		Both	166
		User	490
	1000	System	6
		Both	88
		User	490
	1500	System	9
		Both	80
		User	170.5
3	500	System	5
		Both	33.5
		User	490
	1000	System	5
		Both	56
		User	490
	1500	System	7
		Both	45
		User	166



(a) All directed edges.

(b) Interdicted edges, budget 500.

(c) Interdicted edges, budget 1000.

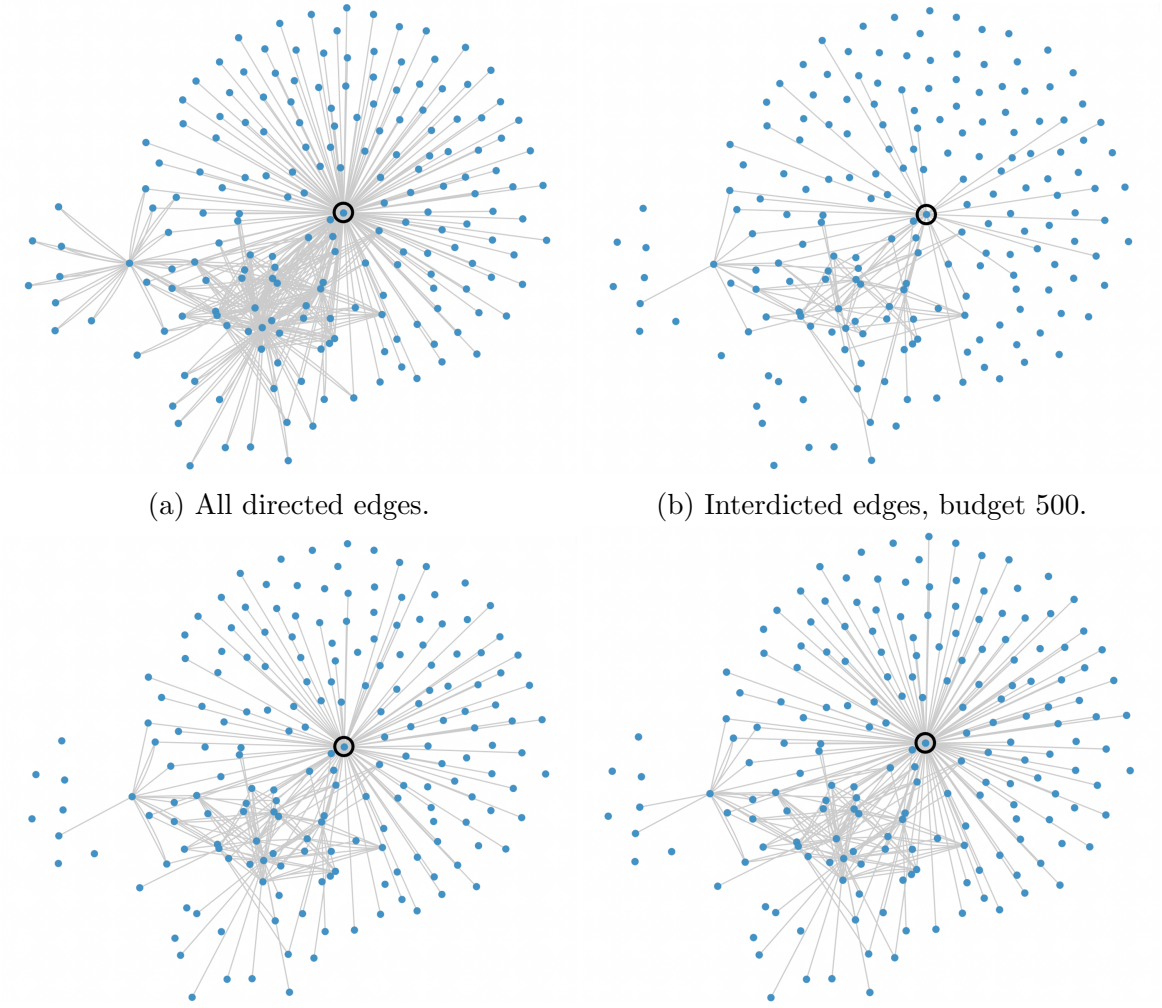
(d) Interdicted edges, budget 1500.

Figure 4.6: Community graph with $|S^c| = 1$ and stochastic users.

system is stochastic, and users deterministically do not self-interdict, the median community size is very low – ranging from 5 to 11. Here, interdiction targets a larger number of smaller communities. On the other hand, when only users are stochastic, and system-initiated interdiction is deterministically effective, the median community size is high – ranging from 166 to 779. In this case, interdiction tends to target larger communities. When both are stochastic, we see a median between these two. This pattern, like the community supergraph structure differences, suggests that centralized, directed action makes more sense when users are stochastic, while widespread action makes more sense when the system is stochastic.

Table 4.6: Characteristics of Interdicted Edges by Test Instance

			Interdicted Edges	
$\|S^c\|$	Budget	Stochastic	Mean Cost	Total Number
1	500	System	3.2	154
		Both	4.7	106
		User	7.4	68
	1000	System	5.5	182
		Both	5	200
		User	7.8	129
	1500	System	6.4	230
		Both	6	250
		User	12.4	121
2	500	System	3.2	154
		Both	4.7	106
		User	7.4	68
	1000	System	5.5	182
		Both	4.9	205
		User	7.9	126
	1500	System	6.5	231
		Both	6	250
		User	12.4	121
3	500	System	3.3	153
		Both	2.7	184
		User	6.3	80
	1000	System	5.5	182
		Both	3.9	255
		User	10.6	94
	1500	System	6.4	230
		Both	5.2	287
		User	11.4	132



(a) All directed edges.

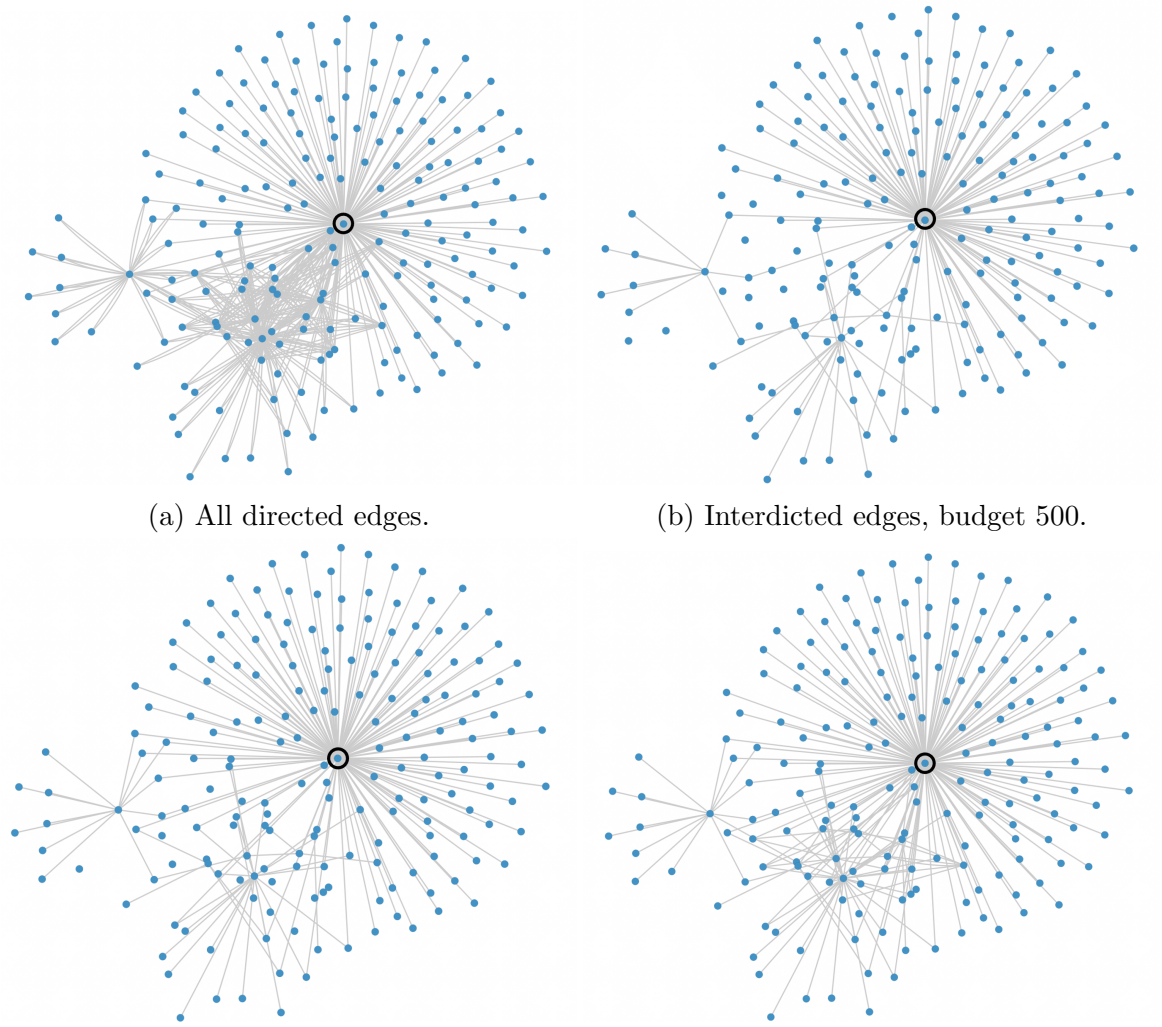
(b) Interdicted edges, budget 500.

(c) Interdicted edges, budget 1000.

(d) Interdicted edges, budget 1500.

Figure 4.7: Community graph with $|S^c| = 1$ and both stochastic users and system.

In Table 4.6, we examine the mean cost of interdicted edges in the community supergraph. Recall that the cost of edge (i, j) is $|E_{ij}|$, the number of edges from community C_i to community C_j in the original social media network. Notice that for fixed $|S^c|$ and budget, no clear pattern exists distinguishing the stochastic system and both stochastic cases, while stochastic users generally always have a higher mean cost and fewer interdicted edges. This suggests that the stochasticity of system-initiated interdiction has a clearer effect on edge cost than the stochasticity of the user. When the system is deterministically effective, in the stochastic users case, there tend to be fewer interdicted edges with higher cost. When the effectiveness of the system is



(a) All directed edges.

(b) Interdicted edges, budget 500.

(c) Interdicted edges, budget 1000.

(d) Interdicted edges, budget 1500.

Figure 4.8: Community graph with $|S^c| = 1$ and stochastic system.

stochastic, regardless of whether users self-interdict, there tend to be more interdicted edges with lower cost. It depends on the test instance how stochastic system and both stochastic compare, however.

Also notice in Table 4.6 that no clear pattern exists for the impact of increasing budget on number of edges interdicted and mean edge cost. For $|S^c| = 3$ and stochastic users, increasing the budget leads to increases in both. At the same time, $|S^c| = 1$ with stochastic users, mean edge cost is 7.4, 7.8, and 12.4, and mean number of edges is 68, 129, and 121. This suggests a greedy heuristic might not perform as well as the

MIP.

4.5 Concluding Remarks

In this chapter, we considered how to mitigate the spread of misinformation or hate speech with limited resources available and uncertainty in intervention effect. We developed a model based on stochastic network interdiction that minimizes the expected maximum flow on a supergraph of a social media network. This supergraph is designed to make equivalent maximum flow and the number of vulnerable users reached in the social media network. We gave an integer program formulation, that uses the dual of maximum flow to create a combined minimization problem, as well as a community-based adaptation for scalability, that we implemented in Python with Gurobi. We derived theoretical results about the nature of fake news mitigation in chained social media networks, with one source node and n target nodes.

In particular, we discovered that a threshold exists for delaying system-initiated interdiction when $n = 2$, based on the probabilities of system-initiated success and the final arc's probability of user self-interdiction, independent of the first arc's probability of user-initiated success. We also found that as system-initiated interdiction is delayed further and further along a path of length n, the absolute difference in expected number of target nodes reached decreases. This allowed us to develop an ϵ-optimal algorithm for single-arc system interdiction of these networks. We demonstrated these results empirically in our numerical studies. In addition, we applied our community-based mixed integer program (FN-Comm) to a large social network. We found that when systemic actions are deterministically effective and users stochastically self-interdict, solutions given by our methods tend to target centralized edges that connect nodes representing large communities. On the other hand, when systemic actions have some probability of failure and users deterministically do not self-interdict, solutions tend to include arcs spread throughout the graph that connect nodes representing

smaller communities. Future research can examine how these solutions perform in the original social media supergraph, under varying assumptions about the stochastic nature of the spread.

CHAPTER V

Conclusion

In this dissertation, we considered three types of crises – evacuations, epidemiological outbreaks, and rampant spread of misinformation on social media.

In Chapter II, we developed two-stage stochastic optimization models for volunteer recruitment in shared-mobility-based evacuation. We introduced an expectation-based stochastic program without probabilistic constraints that can be appropriate in areas with smaller carless populations, where ridesharing could be a feasible option for all evacuees. This model heavily prefers solutions that satisfy all possible demand while minimizing the total cost. The chance-constrained version of the stochastic program can be a better fit in areas with larger carless populations, where large-scale modes of evacuation, such as bus or public transit, are needed. In these settings, ridesharing is a beneficial addition, making use of existing resources to decrease congestion and provide flexible, individualized evacuation to carless populations, especially those that require specialized assistance. At the same time, because ridesharing is more of a helpful service in this setting, it can be useful to find cost-effective solutions that disregard rare, uncharacteristically large or dispersed demand that could skew planning decisions. We also introduced a CVaR-constrained formulation as an alternative risk-averse approach to the chance-constrained formulation.

Our focus was on providing a beneficial, flexible service, able to give special-

ized assistance, to complement the work that has been done on larger-scale transit-dependent evacuation. One potential future research question is how to best integrate large-scale transit and shared mobility in a comprehensive evacuation plan, identifying which populations to assign to which transportation mode and how to time the evacuations, taking into account changing density of traffic and different congestive factors of buses versus personal vehicles.

In Chapter III, we formulated a multi-criteria epidemiological MDP to optimize strategic releases for de-densification of overcrowded facilities during an outbreak. We formulated a cluster-based algorithm to perform large out-of-sample tests of an online sparse sampling policy. We also improved the online sparse sampling algorithm itself, using k-means clustering to create more representative sample nodes in the lookahead tree. These two contributions allowed us to develop a policy that demonstrably improves objective performance over a static benchmark on a large set of multi-epoch trajectories under a range of contact rate assumptions and two priority weights. We also developed epidemiological basis functions for Q learning with linear function approximation, a method with relatively shorter runtimes. Future research can focus on improving the linear function approximation for crime reduction prioritization cases. Additionally, it could be interesting to apply neural networks to the problem, to see how computational runtime and out-of-sample objective performance compare to the online sparse sampling algorithm.

In Chapter IV, we developed a stochastic network interdiction model to minimize the expected number of vulnerable users that receive fake news or hate speech, given a limited intervention budget. These models allow decision makers to optimize interventions in crisis, using resources efficiently and effectively. We demonstrated instances in which system-initiated interdiction should be delayed for path networks with one source node and two target nodes, and developed an ϵ-optimal algorithm for single-source path networks of any length. We developed a community-based trans-

formation of our mixed integer program and tested it on a social network with 23,505 nodes. Future work in this area can consider how solutions in the community-based model perform in the larger social network under a variety of stochastic assumptions, testing the reliability of the solutions when conditions change.

In general, in this thesis, we examined problems within these crises that affect marginalized populations, including carless populations, individuals in isolated, overcrowded facilities, and vulnerable populations on social media. We developed operations research models and solution methods for these problems that included sequential decision making and stochastic modeling of uncertainty. These formulations can help practitioners thoughtfully manage risk, understanding the implications of different policy decisions. As operations researchers develop optimization models and methods, it is important to continue to center marginalized populations, applying these rigorous tools to effectively support the communities in greatest need during crisis.

BIBLIOGRAPHY

BIBLIOGRAPHY

Ahuja, R., T. Magnanti, and J. Orlin (1993), *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall.

Akbarpour, M., C. Cook, A. Marzuoli, S. Mongey, A. Nagaraj, M. Saccarola, P. Tebaldi, S. Vasserman, and H. Yang (2020), Socioeconomic network heterogeneity and pandemic policy response, *University of Chicago, Becker Friedman Institute for Economics Working Paper*, (2020-75).

Albert, R., H. Jeong, and A.-L. Barabási (2000), Error and attack tolerance of complex networks, *Nature*, *406*(6794), 378–382.

Allcott, H., and M. Gentzkow (2017), Social media and fake news in the 2016 election, *Journal of Economic Perspectives*, *31*(2), 211–36.

Arpón, S., T. Homem-de Mello, and B. Pagnoncelli (2018), Scenario reduction for stochastic programs with conditional value-at-risk, *Mathematical Programming*, *170*(1), 327–356.

Baird, L. (1995), Residual algorithms: Reinforcement learning with function approximation, in *Machine Learning Proceedings 1995*, pp. 30–37, Elsevier.

Barak, B., M. Nitzan, N. R. Tannenbaum, and J. Yuval (2020), Optimizing testing policies for detecting COVID-19 outbreaks, *arXiv preprint arXiv:2007.04827*.

Barnhart, C., D. Bertsimas, A. Delarue, and J. Yan (2022), Course scheduling under sudden scarcity: applications to pandemic planning, *Manufacturing & Service Operations Management*, *24*(2), 727–745.

Basciftci, B., X. Yu, and S. Shen (2023), Resource distribution under spatiotemporal uncertainty of disease spread: Stochastic versus robust approaches, *Computers & Operations Research*, *149*, 106,028.

Baycik, N. O., T. C. Sharkey, and C. E. Rainwater (2020), A Markov Decision Process approach for balancing intelligence and interdiction operations in city-level drug trafficking enforcement, *Socio-Economic Planning Sciences*, *69*, 100,700.

Bayram, V. (2016), Optimization models for large scale network evacuation planning and management: A literature review, *Surveys in Operations Research and Management Science*, *21*(2), 63–84.

Ben-Tal, A., B. Do Chung, S. R. Mandala, and T. Yao (2011), Robust optimization for emergency logistics planning: Risk mitigation in humanitarian relief supply chains, *Transportation Research Part B: Methodological*, *45*(8), 1177–1189.

Bertsimas, D., and N. Mundru (2022), Optimization-based scenario reduction for data-driven two-stage stochastic optimization, *Operations Research*.

Bilewicz, M., and W. Soral (2020), Hate speech epidemic. The dynamic effects of derogatory language on intergroup relations and political radicalization, *Political Psychology*, *41*, 3–33.

Birge, J. R., O. Candogan, and Y. Feng (2020), Controlling epidemic spread: Reducing economic losses with targeted closures, *University of Chicago, Becker Friedman Institute for Economics Working Paper*, (2020-57).

Bish, D. R. (2011), Planning for a bus-based evacuation, *OR Spectrum*, *33*(3), 629–654.

Bisset, K. R., X. Feng, M. Marathe, and S. Yardi (2009), Modeling interaction between individuals, social networks and public policy to support public health epidemiology, in *Proceedings of the 2009 Winter Simulation Conference (WSC)*, pp. 2020–2031, IEEE.

Blount, S., A. Galambosi, and S. Yakowitz (1997), Nonlinear and dynamic programming for epidemic intervention, *Applied Mathematics and Computation*, *86*(2-3), 123–136.

Brandeau, M. L., G. S. Zaric, and A. Richter (2003), Resource allocation for control of infectious diseases in multiple independent populations: Beyond cost-effectiveness analysis, *Journal of Health Economics*, *22*(4), 575–598.

Brauer, F. (2008), Compartmental models in epidemiology, in *Mathematical Epidemiology*, pp. 19–79, Springer.

Brown, K. A., A. Jones, N. Daneman, A. K. Chan, K. L. Schwartz, G. E. Garber, A. P. Costa, and N. M. Stall (2020), Association between nursing home crowding and COVID-19 infection and mortality in Ontario, Canada, *JAMA Internal Medicine*.

Budak, C., D. Agrawal, and A. El Abbadi (2011), Limiting the spread of misinformation in social networks, in *Proceedings of the 20th International Conference on World Wide Web*, pp. 665–674.

Caunhye, A. M., X. Nie, and S. Pokharel (2012), Optimization models in emergency logistics: A literature review, *Socio-Economic Planning Sciences*, *46*(1), 4–13.

Charnes, A., and W. W. Cooper (1959), Chance-constrained programming, *Management Science*, *6*(1), 73–79.

Chen, Z., S. Zhang, T. T. Doan, S. T. Maguluri, and J.-P. Clarke (2019), Finite-time analysis of Q-learning with linear function approximation, *arXiv preprint arXiv:1905.11425*.

Churches, T., L. Jorm, et al. (2020), Flexible, freely available stochastic individual contact model for exploring COVID-19 intervention and control strategies: Development and simulation, *JMIR Public Health and Surveillance*, *6*(3), e18,965.

Clauset, A., M. E. Newman, and C. Moore (2004), Finding community structure in very large networks, *Physical Review E*, *70*(6), 066,111.

Cohen, A., and L.-B. Eisen (2022), Reducing jail and prison populations during the COVID-19 pandemic.

Cordeau, J.-F., and G. Laporte (2007), The dial-a-ride problem: models and algorithms, *Annals of Operations Research*, *153*(1), 29–46.

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

Dai, T., and J.-S. Song (2021), Transforming covid-19 vaccines into vaccination: Challenges and opportunities for management scientists, *Health care management science*, *24*(3), 455–459.

Daoust, J.-F. (2020), Elderly people and responses to COVID-19 in 27 countries, *PloS one*, *15*(7), e0235,590.

Deng, Y., H. Jia, S. Ahmed, J. Lee, and S. Shen (2021), Scenario grouping and decomposition algorithms for chance-constrained programs, *INFORMS Journal on Computing*, *33*(2), 757–773.

Du, M., A. Sai, and N. Kong (2021), A data-driven optimization approach for multi-period resource allocation in cholera outbreak control, *European Journal of Operational Research*, *291*(3), 1106–1116.

Duque, D., D. P. Morton, B. Singh, Z. Du, R. Pasco, and L. A. Meyers (2020), Timing social distancing to avert unmanageable COVID-19 hospital surges, *Proceedings of the National Academy of Sciences*, *117*(33), 19,873–19,878.

Enns, E. A., J. J. Mounzer, and M. L. Brandeau (2012), Optimal link removal for epidemic mitigation: A two-way partitioning approach, *Mathematical Biosciences*, *235*(2), 138–147.

EpiModel (2022), Basic DCMs with EpiModel, accessed August 22, 2022.

Fajgelbaum, P., A. Khandelwal, W. Kim, C. Mantovani, and E. Schaal (2020), Optimal lockdown in a commuting network, *Tech. rep.*, National Bureau of Economic Research.

Fanelli, D., and F. Piazza (2020), Analysis and forecast of COVID-19 spreading in China, Italy and France, *Chaos, Solitons & Fractals*, *134*, 109,761.

Farajtabar, M., J. Yang, X. Ye, H. Xu, R. Trivedi, E. Khalil, S. Li, L. Song, and H. Zha (2017), Fake news mitigation via point process based intervention, in *International Conference on Machine Learning*, pp. 1097–1106, PMLR.

FEMA (2006), *Multi-hazard loss estimation methodology: Earthquake model, HAZUS-MH MR2 User Manual*, Department of Homeland Security, FEMA, Washington, DC.

Gan, H.-S., K.-F. Richter, M. Shi, and S. Winter (2016), Integration of simulation and optimization for evacuation planning, *Simulation Modelling Practice and Theory*, *67*, 59–73.

Goerigk, M., K. Deghdak, and V. T'Kindt (2015), A two-stage robustness approach to evacuation planning with buses, *Transportation Research Part B: Methodological*, *78*, 66–82.

Goldman, S., and J. Lightwood (2002), Cost optimization in the SIS model of infectious disease with treatment, *Topics in Economic Analysis & Policy*, *2*(1), 1007.

Gore, A. B., M. E. Kurz, M. J. Saltzman, B. Splitter, W. C. Bridges, and N. J. Calkin (2022), Clemson university's rotational attendance plan during covid-19, *INFORMS Journal on Applied Analytics*, *52*(6), 553–567.

Hagberg, A., P. Swart, and D. S Chult (2008), Exploring network structure, dynamics, and function using NetworkX, *Tech. rep.*, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

He, J., H. Liang, and H. Yuan (2011), Controlling infection by blocking nodes and links simultaneously, in *Internet and Network Economics: 7th International Workshop, WINE 2011, Singapore, December 11-14, 2011. Proceedings 7*, pp. 206–217, Springer.

He, X., G. Song, W. Chen, and Q. Jiang (2012), Influence blocking maximization in social networks under the competitive linear threshold model, in *Proceedings of the 2012 SIAM International Conference on Data Mining*, pp. 463–474, SIAM.

Hou, C., et al. (2020), The effectiveness of quarantine of Wuhan city against the Corona Virus Disease 2019 (COVID-19): A well-mixed SEIR model analysis, *Journal of Medical Virology*, *92*(7), 841–848.

Iosifidis, P., and N. Nicoli (2020), The battle to end fake news: A qualitative content analysis of Facebook announcements on how it combats disinformation, *International Communication Gazette*, *82*(1), 60–81.

Israeli, E., and R. K. Wood (2002), Shortest-path network interdiction, *Networks: An International Journal*, *40*(2), 97–111.

Janjarassuk, U., and J. Linderoth (2008), Reformulation and sampling to solve a stochastic network interdiction problem, *Networks: An International Journal*, *52*(3), 120–132.

Jenness, S. M., S. M. Goodreau, and M. Morris (2018), EpiModel: An R package for mathematical modeling of infectious disease over networks, *Journal of Statistical Software*, *84*.

Jia, H., and S. Shen (2021), Benders cut classification via support vector machines for solving two-stage stochastic programs, *INFORMS Journal on Optimization*, *3*(3), 278–297.

Kearns, M., Y. Mansour, and A. Y. Ng (2002), A sparse sampling algorithm for near-optimal planning in large Markov Decision Processes, *Machine Learning*, *49*(2), 193–208.

Keeling, M. J., and P. Rohani (2011), *Modeling Infectious Diseases in Humans and Animals*, Princeton University Press.

Kempe, D., J. Kleinberg, and É. Tardos (2003), Maximizing the spread of influence through a social network, in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 137–146.

Kermack, W. O., and A. G. McKendrick (1927), A contribution to the mathematical theory of epidemics, *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, *115*(772), 700–721.

Keutchayan, J., J. Ortmann, and W. Rei (2021), Problem-driven scenario clustering in stochastic optimization, *arXiv preprint arXiv:2106.11717*.

Khalil, E. B., P. Le Bodic, L. Song, G. L. Nemhauser, and B. N. Dilkina (2016), Learning to branch in mixed integer programming., in *Proceedings of the thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pp. 724–731.

Kimura, M., K. Saito, and H. Motoda (2009), Blocking links to minimize contamination spread in a social network, *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *3*(2), 1–23.

Kleywegt, A. J., A. Shapiro, and T. Homem-de Mello (2002), The sample average approximation method for stochastic discrete optimization, *SIAM Journal on Optimization*, *12*(2), 479–502.

Koch, D., R. Illner, and J. Ma (2013), Edge removal in random contact networks and the basic reproduction number, *Journal of Mathematical Biology*, *67*(2), 217–238.

Kruber, M., M. E. Lübbecke, and A. Parmentier (2017), Learning when to use a decomposition, in *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pp. 202–210, Springer.

Kulshrestha, A., Y. Lou, and Y. Yin (2014), Pick-up locations and bus allocation for transit-based evacuation planning with demand uncertainty, *Journal of Advanced Transportation*, *48*(7), 721–733.

Legg, M. R., L. K. Nozick, and R. A. Davidson (2010), Optimizing the selection of hazard-consistent probabilistic scenarios for long-term regional hurricane loss estimation, *Structural Safety*, *32*(1), 90–100.

Lemasters, K., E. McCauley, K. Nowotny, and L. Brinkley-Rubinstein (2020), COVID-19 cases and testing in 53 prison systems, *Health & Justice*, *8*(1), 1–6.

Li, A. C., L. Nozick, N. Xu, and R. Davidson (2012), Shelter location and transportation planning under hurricane conditions, *Transportation Research Part E: Logistics and Transportation Review*, *48*(4), 715–729.

Li, M., J. Xu, X. Liu, C. Sun, and Z. Duan (2018), Use of shared-mobility services to accomplish emergency evacuation in urban areas via reduction in intermediate trips: Case study in Xi'an, China, *Sustainability*, *10*(12), 4862.

Li, Y. (2017), Robust vehicle routing in disaster relief and ride-sharing: Models and algorithms, Ph.D. thesis, Binghamton University–SUNY, Binghamton, NY.

Libin, P., A. Moonens, T. Verstraeten, F. Perez-Sanjines, N. Hens, P. Lemey, and A. Nowé (2020), Deep reinforcement learning for large-scale epidemic control, *arXiv preprint arXiv:2003.13676*.

Lim, G. J., M. Rungta, and A. Davishan (2019), A robust chance constraint programming approach for evacuation planning under uncertain demand distribution, *IISE Transactions*, *51*(6), 589–604.

Lindell, M. K. (2013), -evacuation planning, analysis, and management, in *Handbook of Emergency Response*, pp. 156–185, CRC Press.

Litman, T. (2006), Lessons from Katrina and Rita: What major disasters can teach transportation planners, *Journal of Transportation Engineering*, *132*(1), 11–18.

Liu, C., D. M. Aleman, and J. C. Beck (2018), Modelling and solving the senior transportation problem, in *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pp. 412–428, Springer.

Lu, M., Z. Chen, and S. Shen (2018), Optimizing the profitability and quality of service in carshare systems under demand uncertainty, *Manufacturing & Service Operations Management*, *20*(2), 162–180.

Lu, W., L. Liu, F. Wang, X. Zhou, and G. Hu (2020), Two-phase optimization model for ride-sharing with transfers in short-notice evacuations, *Transportation Research Part C: Emerging Technologies*, *114*, 272–296.

Luedtke, J., and S. Ahmed (2008), A sample approximation approach for optimization with probabilistic constraints, *SIAM Journal on Optimization*, *19*(2), 674–699.

Mak, W.-K., D. P. Morton, and R. K. Wood (1999), Monte Carlo bounding techniques for determining solution quality in stochastic programs, *Operations Research Letters*, *24*(1-2), 47–56.

Manjoo, F. (2017), Can Facebook fix its own worst bug?, *The New York Times Magazine*, *24*, 2017.

Manno, A. (2018), Evacuations with I-26 lane reversals for Charleston and most of S.C. coast ordered for Tuesday at noon, Charleston City Paper, `https://www.charlestoncitypaper.com/TheBattery/archives/2018/09/10/evacuations-with-i-26-lane-reversals-for-charleston-and-entire-sc-coast-ordered-for-tuesday-at-noon`.

Mbah, M. L. N., and C. A. Gilligan (2011), Resource allocation for epidemic control in metapopulations, *PLoS one*, *6*(9), e24,577.

Mishra, B. K., A. K. Keshri, Y. S. Rao, B. K. Mishra, B. Mahato, S. Ayesha, B. P. Rukhaiyyar, D. K. Saini, and A. K. Singh (2020), COVID-19 created chaos across the globe: Three novel quarantine epidemic models, *Chaos, Solitons & Fractals*, *138*, 109,928.

Nandi, A. K., and H. R. Medal (2016), Methods for removing links in a network to minimize the spread of infections, *Computers & Operations Research*, *69*, 10–24.

Naoum-Sawaya, J., and J. Y. Yu (2017), Ridesharing for emergency evacuation, *INFOR: Information Systems and Operational Research*, *55*(4), 339–358.

Navabi-Shirazi, M., M. El Tonbari, N. Boland, D. Nazzal, and L. N. Steimle (2022), Multicriteria course mode selection and classroom assignment under sudden space scarcity, *Manufacturing & Service Operations Management*, *24*(6), 3252–3268.

Nguyen, H. T., A. Cano, T. Vu, and T. N. Dinh (2019), Blocking self-avoiding walks stops cyber-epidemics: A scalable GPU-based approach, *IEEE Transactions on Knowledge and Data Engineering*, *32*(7), 1263–1275.

Nowzari, C., V. M. Preciado, and G. J. Pappas (2015), Optimal resource allocation for control of networked epidemic models, *IEEE Transactions on Control of Network Systems*, *4*(2), 159–169.

Radwan, A., A. Hobeika, and D. Sivasailam (1985), Computer simulation model for rural network evacuation under natural disasters, *ITE Journal*, *55*(9), 25–30.

Rawson, T., T. Brewer, D. Veltcheva, C. Huntingford, and M. B. Bonsall (2020), How and when to end the COVID-19 lockdown: An optimization approach, *Frontiers in Public Health*, *8*, 262.

Reluga, T. C. (2009), An SIS epidemiology game with two subpopulations, *Journal of Biological Dynamics*, *3*(5), 515–531.

Renne, J. L., T. W. Sanchez, and T. Litman (2011), Carless and special needs evacuation planning: a literature review, *Journal of Planning Literature*, *26*(4), 420–431.

Rockafellar, R. T., and S. Uryasev (2002), Conditional value-at-risk for general loss distributions, *Journal of banking & finance*, *26*(7), 1443–1471.

Rockafellar, R. T., S. Uryasev, et al. (2000), Optimization of conditional value-at-risk, *Journal of risk*, *2*, 21–42.

Ross, S. M. (2014), *Introduction to stochastic dynamic programming*, Academic Press.

Rujeerapaiboon, N., K. Schindler, D. Kuhn, and W. Wiesemann (2018), Scenario reduction revisited: Fundamental limits and guarantees, *Mathematical Programming*, pp. 1–36.

Saadatseresht, M., A. Mansourian, and M. Taleai (2009), Evacuation planning using multiobjective evolutionary optimization approach, *European Journal of Operational Research*, *198*(1), 305–314.

Sadri, A. M., S. V. Ukkusuri, P. Murray-Tuite, and H. Gladwin (2014), Analysis of hurricane evacuee mode choice behavior, *Transportation Research Part C: Emerging Technologies*, *48*, 37–46.

Savelsbergh, M. W., and M. Sol (1995), The general pickup and delivery problem, *Transportation Science*, *29*(1), 17–29.

Saxena, A., P. Saxena, and H. Reddy (2022), Fake news propagation and mitigation techniques: A survey, *Principles of Social Networking: The New Horizon and Emerging Challenges*, pp. 355–386.

Shu, K., A. Sliva, S. Wang, J. Tang, and H. Liu (2017), Fake news detection on social media: A data mining perspective, *ACM SIGKDD explorations newsletter*, *19*(1), 22–36.

Silva, P. C., P. V. Batista, H. S. Lima, M. A. Alves, F. G. Guimarães, and R. C. Silva (2020), COVID-ABS: An agent-based model of COVID-19 epidemic to simulate health and economic effects of social distancing interventions, *Chaos, Solitons & Fractals*, *139*, 110,088.

Smith, D. R., A. Duval, K. B. Pouwels, D. Guillemot, J. Fernandes, B.-T. Huynh, L. Temime, and L. Opatowski (2020), Optimizing COVID-19 surveillance in long-term care facilities: A modelling study, *BMC Medicine*, *18*(1), 1–16.

Smith, J. C., and Y. Song (2020), A survey of network interdiction models and algorithms, *European Journal of Operational Research*, *283*(3), 797–811.

South Carolina Department of Transportation (2018), Traffic counts, https://www.scdot.org/travel/travel-trafficdata.aspx.

Tanaka, A., N. Hata, N. Tateiwa, and K. Fujisawa (2017), Practical approach to evacuation planning via network flow and deep learning, in *2017 IEEE International Conference on Big Data (Big Data)*, pp. 3368–3377, IEEE.

Tanınmış, K., N. Aras, İ. K. Altınel, and E. Güney (2020), Minimizing the misinformation spread in social networks, *Iise Transactions*, *52*(8), 850–863.

Thunström, L., S. C. Newbold, D. Finnoff, M. Ashworth, and J. F. Shogren (2020), The benefits and costs of using social distancing to flatten the curve for COVID-19, *Journal of Benefit-Cost Analysis*, pp. 1–27.

Toth, P., and D. Vigo (2002), *The vehicle routing problem*, SIAM.

Tseng, T.-G., H.-L. Wu, H.-C. Ku, and C.-J. Tai (2020), The impact of the COVID-19 pandemic on disabled and hospice home care patients, *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences*.

Tweedie, S. W., J. R. Rowland, S. J. Walsh, R. P. Rhoten, and P. I. Hagle (1986), A methodology for estimating emergency evacuation times, *The Social Science Journal*, *23*(2), 189–204.

United States Census Bureau (2018a), 2018 ACS 5-Year Estimates Data Profiles: ACS Demographic and Housing Estimates, https://data.census.gov/cedsci/table?q=age&lastDisplayedRow=93&table=DP05&tid=ACSDP5Y2018.DP05&t=Age%20and%20Sex&layer=censustract&vintage=2017&mode=customize&hidePreview=true&cid=DP05_0001E&moe=false&g=0500000US45019.140000.

United States Census Bureau (2018b), 2018 ACS 5-Year Estimates Subject Profiles: Median Income in the Past 12 Months (in 2018 Inflation-Adjusted Dollars), https://data.census.gov/cedsci/table?q=income&g=0500000US45019.140000&hidePreview=false&table=S1903&tid=ACSST5Y2018.S1903&lastDisplayedRow=23&vintage=2018&layer=censustract&cid=S0101_C01_001E&t=Income%20%28Households,%20Families,%20Individuals%29.

Urbina, E., and B. Wolshon (2003), National review of hurricane evacuation plans and policies: a comparison and contrast of state practices, *Transportation Research Part A: Policy and Practice*, *37*(3), 257–275, doi:10.1016/S0965-8564(02)00015-0.

Van Dorn, A., R. E. Cooney, and M. L. Sabin (2020), COVID-19 exacerbating inequalities in the US, *Lancet (London, England)*, *395*(10232), 1243.

von Stackelberg, H., and A. Peacock (1952), *The Theory of the Market Economy*, William Hodge.

Wang, L. (2020), A two-stage stochastic programming framework for evacuation planning in disaster responses, *Computers & Industrial Engineering*, p. 106458.

Wang, L., L. Yang, Z. Gao, S. Li, and X. Zhou (2016), Evacuation planning for disaster responses: A stochastic programming framework, *Transportation Research Part C: Emerging Technologies*, *69*, 150–172.

Wang, X., K. Deng, J. Li, J. X. Yu, C. S. Jensen, and X. Yang (2018), Targeted influence minimization in social networks, in *Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part III 22*, pp. 689–700, Springer.

Wang, X., R. F. Pasco, Z. Du, M. Petty, S. J. Fox, A. P. Galvani, M. Pignone, S. C. Johnston, and L. A. Meyers (2020), Impact of social distancing measures on coronavirus disease healthcare demand, central Texas, USA, *Emerging Infectious Diseases*, *26*(10), 2361.

Wollmer, R. (1964), Removing arcs from a network, *Operations Research*, *12*(6), 934–940.

Wong, R. T. (2008), Vehicle routing for small package delivery and pickup services, in *The Vehicle Routing Problem: Latest Advances and New Challenges*, pp. 475–485, Springer.

Yaesoubi, R., and T. Cohen (2011), Generalized Markov models of infectious disease spread: A novel framework for developing dynamic health policies, *European Journal of Operational Research*, *215*(3), 679–687.

Yin, X., İ. E. Büyüktahtakın, and B. P. Patel (2023), Covid-19: Data-driven optimal allocation of ventilator supply under uncertainty and risk, *European journal of operational research*, *304*(1), 255–275.

Yu, M., and S. Shen (2020), An integrated car-and-ride sharing system for mobilizing heterogeneous travelers with application in underserved communities, *IISE Transactions*, *52*(2), 151–165.

Yusoff, M., J. Ariffin, and A. Mohamed (2008), Optimization approaches for macroscopic emergency evacuation planning: a survey, in *2008 International Symposium on Information Technology*, vol. 3, pp. 1–7, IEEE.

Zaric, G. S., and M. L. Brandeau (2001), Resource allocation for epidemic control over short time horizons, *Mathematical Biosciences*, *171*(1), 33–58.

Zaric, G. S., and M. L. Brandeau (2002), Dynamic resource allocation for epidemic control in multiple populations, *Mathematical Medicine and Biology*, *19*(4), 235–255.

Zhang, J., L. Cui, Y. Fu, and F. B. Gouza (2018), Fake news detection with deep diffusive network model, *arXiv preprint arXiv:1805.08751*.

Zhang, Y., M. Lu, and S. Shen (2021), On the values of vehicle-to-grid electricity selling in electric vehicle sharing, *Manufacturing & Service Operations Management*, *23*(2), 488–507.

Zhou, X., and R. Zafarani (2020), A survey of fake news: Fundamental theories, detection methods, and opportunities, *ACM Computing Surveys (CSUR)*, *53*(5), 1–40.

Zhou, X., A. Jain, V. V. Phoha, and R. Zafarani (2020), Fake news early detection: A theory-driven model, *Digital Threats: Research and Practice*, *1*(2), 1–25.

Zoraster, R. M., et al. (2010), Vulnerable populations: Hurricane Katrina as a case study, *Prehospital and disaster medicine*, *25*(1), 74–78.