8-31-2021

Multi-stage stochastic optimization and reinforcement learning for forestry epidemic and covid-19 control planning

Sabah Bushaj

New Jersey Institute of Technology

ABSTRACT

MULTI-STAGE STOCHASTIC OPTIMIZATION AND REINFORCEMENT LEARNING FOR FORESTRY EPIDEMIC AND COVID-19 CONTROL PLANNING

by
Sabah Bushaj

This dissertation focuses on developing new modeling and solution approaches based on multi-stage stochastic programming and reinforcement learning for tackling biological invasions in forests and human populations. Emerald Ash Borer (EAB) is the nemesis of ash trees. This research introduces a multi-stage stochastic mixed-integer programming model to assist forest agencies in managing emerald ash borer insects throughout the U.S. and maximize the public benefits of preserving healthy ash trees. This work is then extended to present the first risk-averse multi-stage stochastic mixed-integer program in the invasive species management literature to account for extreme events. Significant computational achievements are obtained using a scenario dominance decomposition and cutting plane algorithm. The results of this work provide crucial insights and decision strategies for optimal resource allocation among surveillance, treatment, and removal of ash trees, leading to a better and healthier environment for future generations.

This dissertation also addresses the computational difficulty of solving one of the most difficult classes of combinatorial optimization problems, the Multi-Dimensional Knapsack Problem (MKP). A novel Two-Dimensional (2D) deep reinforcement learning (DRL) framework is developed to represent and solve combinatorial optimization problems focusing on MKP. The DRL framework trains different agents for making sequential decisions and finding the optimal solution while still satisfying the resource constraints of the problem. This is the first DRL model of its kind where a 2D environment is formulated, and an element of the DRL solution matrix represents an item of the MKP. The DRL framework shows that it can solve medium-sized and large-sized instances at least 45 and 10 times faster in CPU solution time, respectively, with a maximum solution gap of 0.28% compared to the solution performance of CPLEX. Applying this methodology, yet another recent epidemic problem is tackled, that of COVID-19. This research investigates a reinforcement learning approach tailored with an agent-based simulation model to simulate the disease growth and optimize decision-making during an epidemic. This framework is validated using the COVID-19 data from the Center for Disease Control and Prevention (CDC). Research results provide important insights into government response to COVID-19 and vaccination strategies.

MULTI-STAGE STOCHASTIC OPTIMIZATION AND REINFORCEMENT LEARNING FOR FORESTRY EPIDEMIC AND COVID-19 CONTROL PLANNING

by
Sabah Bushaj

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Industrial Engineering

Department of Mechanical and Industrial Engineering

August 2021

APPROVAL PAGE

MULTI-STAGE STOCHASTIC OPTIMIZATION AND REINFORCEMENT LEARNING FOR FORESTRY EPIDEMIC AND COVID-19 CONTROL PLANNING

Sabah Bushaj

Dr. İ. Esra Büyüktahtakın Toy, Dissertation Advisor — Date
Associate Professor of Mechanical and Industrial Engineering, NJIT

Dr. Layek Abdel-Malek, Committee Member — Date
Professor of Mechanical and Industrial Engineering, NJIT

Dr. Sanchoy K. Das, Committee Member — Date
Professor of Mechanical and Industrial Engineering, NJIT

Dr. Selina Cai, Committee Member — Date
Associate Professor of Mechanical and Industrial Engineering, NJIT

Dr. Ecevit Bilgili, Committee Member — Date
Associate Professor of Chemical and Materials Engineering, NJIT

BIOGRAPHICAL SKETCH

Author:	Sabah Bushaj
Degree:	Doctor of Philosophy
Date:	August 2021

Undergraduate and Graduate Education:

- Doctor of Philosophy in Industrial Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2021
- Master of Science in Computer Science,
  Epoka University, Tirane, Albania, 2016
- Bachelor of Science in Business Informatics,
  Epoka University, Tirane, Albania, 2014

Publications:

S. Bushaj, İ. E. Büyüktahtakın, R.G. Haight, and D. Yemshanov. Optimizing Surveillance and Management of Emerald Ash Borer in Urban Environments. *Natural Resource Modelling*, 2020.

S. Bushaj, İ. E. Büyüktahtakın,and R.G. Haight. Risk-Averse Multi-Stage Stochastic Optimization for Surveillance and Operations Planning of a Forest Insect Infestation. *Second revision submitted to European Journal of Operational Research*, 2021.

S. Bushaj and İ. E. Büyüktahtakın. Deep Reinforcement Learning Approach for Solving Multi-Dimensional Knapsack Problem. *Under preparation*, 2021.

S. Bushaj and İ. E. Büyüktahtakın. A Simulation-Deep Reinforcement Learning (SIRL) Optimization Approach to Controlling the COVID-19 Pandemic. *Under preparation*, 2021.

S. Onal, **S. Bushaj**, İ. E. Büyüktahtakın, Jennifer Smith, Gregory R. Houseman. A Gaussian Dispersal Approach to Capture Long-Term and Long-Distance Dispersal Through Simulation-Optimization. *Under preparation*, 2021.

Presentations:

S. Bushaj, İ. E. Büyüktahtakın, D. Yemshanov and R. G. Haight. *Optimizing Surveillance and Management of Emerald Ash Borer in Urban Environments. Modeling and Optimization: Theory and Applications*, Bethlehem, PA 2019.

S. Bushaj, İ. E. Büyüktahtakın, D. Yemshanov and R. G. Haight. *Optimizing Search and Control of Emerald Ash Borer in Urban Environments*, *Institute for Operations Research and the Management Sciences Annual Meeting.* Seattle, WA 2019.

S. Bushaj, İ. E. Büyüktahtakın, D. Yemshanov and R. G. Haight. *Multistage Stochastic Programming to Optimize Surveillance and Management of Emerald Ash Borer in Urban Environments. Graduate Research Day at New Jersey Institute of Technology*, Newark, NJ 2019.

S. Bushaj, İ. E. Büyüktahtakın and R. G. Haight. *Risk-Averse Multi-Stage Stochastic Problem Formulation. Systems Optimization and Data Analytics Lab (SODAL) Meeting with Minneapolis Forest Services*, Online 2019.

S. Bushaj and İ. E. Büyüktahtakın. *A Multi-Stage Stochastic Programming Approach to the Optimal Surveillance and Control of Emerald Ash Borer in Cities. SODAL Meeting with NJ Forest Services*, Online 2019.

S. Bushaj, İ. E. Büyüktahtakın and R. G. Haight. *A Risk-Averse Multistage Stochastic Program to Optimize Search and Control of Emerald Ash Borer in Cities. Institute for Operations Research and the Management Sciences Annual Virtual Meeting*, Online 2020.

S. Bushaj, İ. E. Büyüktahtakın, X. Yin. *A Simulation-Deep Reinforcement Learning (SiRL) Optimization Approach to Controlling the COVID-19. Institute for Operations Research and the Management Sciences Healthcare Conference*, Online 2021.

X. Yin, İ. E. Büyüktahtakın, **S. Bushaj**. *An Integrated Simulation-Optimization Algorithmic Framework to Vaccine Distribution for Controlling the COVID-19. Institute for Operations Research and the Management Sciences Healthcare Conference*, Online 2021.

S. Bushaj, İ. E. Büyüktahtakın. *A Deep Reinforcement Learning Approach for Solving Multi-Dimensional Knapsack Problem. Mixed-Integer Programming Workshop 2021*, Online 2021.

Krenaria ime nuk qëndron te kjo diplomë, krenaria ime jeni ju prindërit e mi të shtrenjtë!

My pride is not in this dissertation, my pride are you dear parents!

ACKNOWLEDGMENT

I would like to express my most sincere gratitude to my advisor, Dr. İ. Esra Büyüktahtakın Toy, for guiding me through every step of the way. Her vision and motivation served as an endless source of inspiration for me. With her guidance on methodology to carry out research I feel ready to follow in her footsteps and contribute to scientific society. It has been a great privilege for me to study and work under her guidance.

I wish to extend my thanks to the committee members. With their expertise and direction, they helped me stay on track dig deeper where it mattered. Thank you, Dr. İ. Esra Büyüktahtakın Toy, Dr. Sanchoy K. Das, Dr. Layek Abdel-Malek, Dr. Selina Cai, and Dr. Ecevit Bilgili for serving on my committee.

I also would like to acknowledge Dr. Robert Haight from the United States Department of Agriculture and Forest Service for his help and expertise offered during my research.

I gratefully acknowledge the support of the U.S. Department of Agriculture, Forest Service, Northern Research Station Joint Venture Agreement No. 16-JV-11242309-109 and the National Science Foundation CAREER Award co-funded by the CBET/ENG Environmental Sustainability program and the Division of Mathematical Sciences in MPS/NSF under Grant No. CBET-1554018.

I also would like to thank my peers at the Systems Optimization and Data Analytics Lab (SODAL), Sevilay Onal, Xuecheng Yin, and Dogacan Yilmaz, for their collaboration and promoting a fun and productive working environment.

Finally, I am indebted to my family and friends. This thesis would have remained a dream had it not been for all the support I have been given. Particularly, I owe my deepest gratitude to my parents, Hazir and Kimete, and my wife, Arjeta, for providing me the means and the emotional power to follow through any obstacle.

TABLE OF CONTENTS

Chapter **Page**

1 INTRODUCTION 1

1.1 Background in Methodology 5

1.2 Motivation and Contributions 9

1.3 Summary of Research Objectives and Accomplishments 11

1.4 Organization of the Dissertation 15

2 OPTIMIZING SURVEILLANCE AND MANAGEMENT OF EMERALD ASH BORER IN URBAN ENVIRONMENTS 16

2.1 Introduction 16

2.1.1 Literature Review 19

2.1.2 Key Contributions of the Chapter 20

2.2 Optimization Model 21

2.2.1 Problem Definition 22

2.2.2 Scenario Tree 24

2.2.3 Algorithm for Scenario Tree Probability 26

2.2.4 Notation 27

2.2.5 Mathematical Model 30

2.3 Model Application and Data 37

2.4 Results 40

2.4.1 Optimal Management 40

2.4.2 Effect of Surveillance Timing on Net Benefits 44

2.4.3 Budget Allocation and Priority of Actions 45

2.4.4 Is treatment a choice? 48

2.4.5 Impact of Surveillance Efficiency 49

2.5 Discussion and Conclusions 52

TABLE OF CONTENTS
(Continued)

Chapter Page

3 RISK-AVERSE MULTI-STAGE STOCHASTIC OPTIMIZATION FOR SURVEILLANCE AND OPERATIONS PLANNING OF A FOREST INSECT INFESTATION 55

3.1 Introduction 55

3.2 Literature Review 57

3.2.1 Risk-Averse Stochastic Programming 57

3.2.2 EAB Control and Risk-Averse Forest Management Planning . 60

3.2.3 Key Contributions 61

3.3 Risk-Averse MSS-MIP Framework 64

3.3.1 General Formulation of MSS-MIP 64

3.3.2 CVaR, the Risk Averse Measure of Choice 65

3.3.3 Time-Consistent Mean-Risk MSS-MIP 67

3.4 RA-MSS-MIP for the Surveillance and Operations Planning of EAB . 72

3.4.1 Notation 72

3.4.2 Mathematical Model 74

3.5 Scenario Dominance Decomposition 80

3.5.1 Sub-additivity and Upper Bound 82

3.5.2 Lower Bound 83

3.5.3 Scenario Dominance 84

3.5.4 Cuts based on Scenario Dominance 86

3.5.5 Cutting-Plane Algorithms 93

3.6 Computational Experiments 94

3.6.1 Implementation Details 94

3.6.2 Instance Generation and Test Data 96

3.6.3 Results 96

3.7 Concluding Remarks 108

TABLE OF CONTENTS
(Continued)

Chapter **Page**

4 A DEEP REINFORCEMENT LEARNING APPROACH FOR SOLVING MULTI-DIMENSIONAL KNAPSACK PROBLEM 111

4.1 Introduction 111

4.1.1 Notations 112

4.2 Related Work 113

4.2.1 Key Contributions 117

4.3 Multi-Dimensional Knapsack Problem Formulation 119

4.4 Deep Reinforcement Learning Knapsack Model 120

4.4.1 Heuristic Transformation 120

4.4.2 K-means Algorithm and Initial Solution 122

4.4.3 DRL Model 124

4.4.4 One-dimensional Knapsack Environment 125

4.4.5 Two-dimensional Knapsack Environment 128

4.4.6 Main Algorithm 129

4.4.7 Generalization to Larger Instances 132

4.5 Experiments 133

4.5.1 Instance Generation and Implementation 133

4.5.2 Implementation Details 134

4.5.3 Results 135

4.6 Discussion and Future Work 147

5 A SIMULATION-DEEP REINFORCEMENT LEARNING (SIRL) APPROACH FOR EPIDEMIC OPTIMIZATION AND CONTROL . . . 148

5.1 Introduction 148

5.2 Related Work 150

5.2.1 Key Contributions 156

5.3 Simulation Environment 157

5.4 DRL Environment 161

TABLE OF CONTENTS
(Continued)

Chapter **Page**

5.4.1 Episode and States 161

5.4.2 Multi-Objective Reward Function 162

5.4.3 Actions or Intervention Measures 163

5.5 Integrated Simulation-RL 165

5.5.1 Training Algorithm 166

5.6 Experiments 168

5.6.1 Data Gathering 168

5.6.2 Results 174

5.7 Discussion and Future Work 182

6 SUMMARY AND FUTURE RESEARCH DIRECTIONS 184

6.1 Summary of Contributions 184

6.2 Future Research Directions 185

APPENDIX A FURTHER NOTES CHAPTER 2 187

A.1 Two-Stage Example of Possible Scenarios 187

A.2 Scenario Probability Algorithm 188

APPENDIX B FURTHER NOTES CHAPTER 3 191

B.1 Model Description and Assumptions 191

B.1.1 Scenario Tree Generation 193

B.2 Time-Consistent Mean-Risk MSS-MIP 195

B.3 Case Study Data 196

B.4 Gap Improvement Calculation for Sdc and Ssdc Cuts 199

B.5 Effect of Risk Parameters on Time Complexity 201

APPENDIX C FURTHER NOTES ON CHAPTER 5 203

C.1 Notations and Abbreviations 203

REFERENCES 206

LIST OF TABLES

Table Page

2.1 Net Benefits in the Optimal Management (Action) and No-Action Solutions for Scenario H-H-H-H-H, H-H-L-L-L, L-L-H-H-H, and L-L-L-L-L . . . 44

2.2 Costs of Surveillance, Treatment, and Removal for the Scenarios with Different Timing of the Survey Actions . . . 45

2.3 Net Benefit Values in Optimal Treatment vs No-Treatment Under Various Scenarios . . . 49

2.4 Net Benefit Values in the Solutions Using Branch Sampling and Sticky Traps . . . 52

3.1 Experiment Results for CPX, SDC, and SSDC Under Different Infestation Patterns . . . 98

3.2 Experiment Results for CPX, SDC, and SSDC Under Different Landscape Sizes . . . 100

3.3 Comparison of Objective Values, Expected Benefit and Risk, and Costs for the Risk-Neutral, Low, Moderate, and High Risk-Averse Models . 101

3.4 Risk Values for Five Scenario Realizations with Different Surveillance Frequency Under Various Risk Levels* . . . 106

3.5 Comparison of Objective Values, Expected Benefit and Risk, and Costs for the Risk-Neutral, Low, Moderate, and High Risk-Averse Models . 108

4.1 DRL, K-means, and Heuristic Parameters . . . 135

4.2 Comparison of DRL Algorithms and CPLEX Performances . . . 141

4.3 Comparison of DRL Algorithms with CPLEX for Test Instances with $R = 25$ . . . 142

4.4 Comparison of DRL Algorithms with CPLEX for Test Instances with $R = 100$ . . . 143

4.5 Three Levels of Partial Prediction for Four DRL Algorithms and CPLEX 146

5.1 Paired T-Test Analysis Comparing the Bi-Weekly Compartmental Data from the NVM Simulation with the Actual Data from the CDC . . . 172

5.2 Paired T-Test Analysis Comparing the Bi-Weekly Compartmental Data from the AVM Simulation with the Actual Data from the CDC . . . 173

LIST OF TABLES
(Continued)

Table **Page**

5.3 Comparison of different tuning parameters in the reward function for age-based vaccination. 180

5.4 Comparison of Vaccine Distribution Among Different Age Groups and Compartmental Values for Each Model 181

A.1 All Possible Scenarios for a Two-Stage Formulation 188

B.1 Calculations for Integrality Gap Improvement by SDC and SSDC 200

B.2 Calculations for Integrality Gap Improvement by SDC and SSDC 201

B.3 Time Complexity of Risk-Neutral, CPX, SDC, and SSDC Models for Different Risk Parameters 202

LIST OF FIGURES

Figure **Page**

2.1 Transitions between the infestation levels. 23

2.2 An example transition diagram of the ash tree population with over a 5-period planning horizon under a particular surveillance regime (i.e., surveys in years 2 and 5 and no surveillance in years 1, 3 and 4). X-axis denotes time periods and y-axis represents tree infestation levels. Node S denotes the susceptible trees and nodes 1, 2, and 3 denote the infestation levels 1-3 (asymptomatic, symptomatic and dead trees). In each stage, uncolored nodes represent the infestation levels for which the actual number of infested trees is unknown because no surveillance was made. Light shaded (green) nodes depict the detected susceptible trees; dark-shaded (red) nodes depict trees at infestation levels 1-3 after surveillance; and black nodes represent the infested dead trees that must be removed. Dashed green lines show the number of treated trees that become temporarily immune to an infestation and are moved to a pool of susceptible trees; dotted lines show the transition in tree infestation level from one period to another; and bold arrows show infested dead trees that are removed. *Source: Adapted from Kıbış et al. [2021].* . . . 25

2.3 Multi-stage scenario tree. Terms p_t^H and p_t^L denote the realization probabilities of high (H) and low (L) levels of infestation after surveillance; p_t^M denotes a default realization of medium infestation level (M) without surveillance. Black circles represent nodes with decisions after the surveillance; white circles depict nodes without surveillance; arcs leaving black and white circles depict possible realizations of the estimated beliefs about the number of susceptible and infested trees; red arrows depict realizations of high infestation levels; green arrows depict realizations of low infestation levels; yellow arrows depict anticipated levels of infestation without surveillance based on the initial belief. Squares $D_{t,s}$ depict treatment and removal decisions. *Source: Adapted from Kıbış et al. [2021].* 26

2.4 Estimating the probabilities of the scenario occurrence for a 3-period example using Algorithm 1. Bold red lines show realizations of high infestation level, dashed green lines show realizations of low infestation level after surveillance, and dotted black lines show realizations of the expected medium infestation level without surveillance. The numbers between lines show the probabilities of scenario realizations. $\pi_1 - \pi_{3t}$ depict the probabilities for each scenario ω. 27

LIST OF FIGURES (Continued)

Figure Page

2.5 Host tree density map for a case study area in Winnipeg, Canada. Square outline delineates the neighborhood area around the current EAB infestations. 38

2.6 Treatment, removal, and surveillance cost for low and high realizations over 5 years for four different scenarios. Scenario 0 is H-H-H-H-H, scenario 4 is H-H-H-L-L, scenario 117 is L-L-L-H-H and lastly, scenario 121 is L-L-L-L-L, where L and H stand for low and high realization, respectively. 41

2.7 Total number of infested trees under no action, infested trees under optimal action, treated, and removed trees for each period for scenario H-H-H-H-H (high infestation level detected in all periods) over planning periods 1-4. 43

2.8 Total number of infested trees under no action, infested trees under optimal action, treated, and removed trees for each period for scenario L-L-L-L-L (low infestation level detected in all periods) over planning periods 1-4. 43

2.9 Surveillance, treatment, and removal costs for different budget levels under worst-case scenario H-H-H-H-H with continuous surveillance over the planning horizon and the detected high level of infestation. 46

2.10 Total number of infested, treated, and removed trees over time under scenario 0 with \$1.5M (a) and \$2M (b) budget. 48

2.11 Objective function values for scenario 0 (H-H-H-H-H) using branch sampling and sticky trap methods with \$2M budget. 50

2.12 Surveillance, treatment, and removal cost comparison between branch sampling and sticky trap methods for scenario 0 (H-H-H-H-H) with \$2M budget 50

2.13 Total number of infested, treated and removed trees over time for scenario 0 with budget \$2M: a) using branch sampling; b) using sticky traps. . 51

3.1 Algorithms for sdc (3.1a) and ssdc (3.1b) 94

3.2 The CDF of benefit values of all scenarios for risk-neutral and risk-averse models. 103

3.3 Expected net benefit versus CVaR for each combination of risk parameters α and λ under both risk-neutral ($\lambda = 0$) and the risk-averse cases ($\lambda > 0$).105

LIST OF FIGURES
(Continued)

Figure **Page**

3.4 Expected profit for the top 100 × α% worst-case scenarios for different values of λ under both risk-neutral ($\lambda = 0$) and the risk-averse cases ($\lambda > 0$). 106

4.1 Reverting the items based on the worthiness ratios. 122

4.2 One-dimensional vector representation of our problem. 126

4.3 Two-dimensional matrix representation of the state space. 128

4.4 Training DRL flowchart. 130

4.5 Two-dimensional DRL Environment for 100 items and 100 constraints. . 133

4.6 Two-dimensional DRL Environment for 400 items and 400 constraints. . 133

4.7 Two-dimensional DRL Environment for 900 items and 900 constraints. . 134

4.8 Percent gap for small instances. 137

4.9 Violated constraints for small instances. 137

4.10 Percent gap for medium instances. 137

4.11 Violated constraints for medium instances. 137

4.12 Percent gap for large instances. 137

4.13 Violated constraints for large instances. 137

4.14 Training gap and rewards for each episode where a negative reward represents a positive contribution to reducing the minimization objective function. 138

5.1 Covasim disease progression, compartments, and final outcomes [Kerr et al., 2020]. 158

5.2 Covasim disease progression, compartments, and final outcomes in the extended model. 160

5.3 The SiRL Framework, which is an agent-based simulation integrated at the heart of a DRL framework. 166

5.4 Agent-based simulation and DRL agent information exchange between simulation periods. 167

5.5 COVID-19 timeline from April 1, 2020, to June 30, 2020, created based on the information provided in Thebault et al. [2021]. 169

LIST OF FIGURES (Continued)

Figure **Page**

5.6 Comparison of the Covasim agent-based simulation data and CDC data for the state of New Jersey for the period from March 1, 2020, to June 30, 2020. The x-axis shows the timeline where each dot on the trend lines corresponds to a decision. The y-axis shows the absolute value of the difference between the real number of individuals in each compartment (S, I, H, C, D, R, T) at a point in time and the number of individuals estimated from the simulation in that compartment $(S_s, I_s, H_s, C_s, D_s, R_s, T_s)$. For example, trend line $|S - S_s|$ represents the absolute difference between the real susceptible proportion of the population (S) and simulated susceptible proportion (S_s) at bi-weekly dates starting March 1 to June 30. Similarly, the trend lines for each compartment are plotted. 171

5.7 Comparison of the Covasim agent-based simulation data and CDC date for the state of New Jersey for the period from December 15, 2020, to April 15, 2021. The x-axis shows the timeline where each dot on the trend lines corresponds to a decision. The y-axis shows the absolute value of the difference between the real number of individuals in a compartment at a point in time and the number of individuals estimated in that compartment from the simulation. For example, trend line $|H - H_s|$ represents the absolute value of the difference between the hospitalized individuals in New Jersey and the hospitalized individuals estimated by the simulation at bi-weekly dates starting December 15, to April 15. Similarly, the trend lines for each compartment are plotted. 173

5.8 Reward for each training epoch for the **NVM** model. 174

5.9 Reward for each training epoch for the **AVM** model. 175

5.10 Comparison of government actions and DRL agent actions during the first four months of the COVID-19 pandemic in the U.S. Above the x-axis we describe the government actions and below the x-axis the DRL agent suggestions are shown. 176

5.11 Comparison of government and DRL agent actions for the first 4 months after vaccine was introduced for COVID-19 using an age group vaccination strategy. Above the x-axis, we describe the government actions, and below x-axis, the DRL agent suggestions are shown. . . 177

5.12 COVID-19 timeline allowing all age groups access to vaccination. Above the x-axis, we describe the government actions and below x-axis, the DRL agent suggestions are shown. 178

5.13 Comparison of the economic index between NJ CDC Data and simulation data using NVM. 178

LIST OF FIGURES (Continued)

Figure **Page**

5.14 Comparison of the economic index between NJ CDC Data and simulation data using AVM and RVM. 179

B.1 Multi-stage scenario tree. Terms p_t^H and p_t^L denote the realization probabilities of high (H) and low (L) levels of infestation after surveillance; p_t^M denotes a default realization of medium infestation level (M) without surveillance. Black circles represent nodes with decisions after the surveillance; white circles depict nodes without surveillance; arcs leaving black and white circles depict possible realizations of the estimated beliefs about the number of susceptible and infested trees; red arrows depict realizations of high infestation levels; green arrows depict realizations of low infestation levels; yellow arrows depict anticipated levels of infestation without surveillance based on the initial belief. Squares $D_{t,s}$ depict treatment and removal decisions. *Source: Adapted from Kıbış et al. [2021], Bushaj et al. [2021b].* 194

B.2 New Jersey ash tree population 199

B.3 New Jersey EAB detection data [USDA, 2018] 199

LIST OF ABBREVIATIONS

AVM:	Age-Based Vaccination Model
A2C:	Advantage Actor-Critic.
A3C:	Asynchronous Advantage Actor-Critic.
ACKTR:	Actor-Critic with Kronecker-Factored Trust Region.
BIP:	Binary Integer Programming.
CISM	Center for Invasive Species Management.
CDC:	Center for Disease Control and Prevention
COP:	Combinatorial Optimization Problem.
CVaR:	Conditional Value at Risk.
DDQN:	Double Deep Q-Network.
DQN:	Deep Q-Network.
DRL:	Deep Reinforcement Learning.
DP:	Dynamic Programming.
EUA:	Emergency Use Authorization.
FDA:	Food and Drug Administration.
FMSTS:	Fitting for Minimizing SubTree Size.
EAB:	Emerald Ash Borer.
ECRM:	Expected Conditional Risk Measure.
ECSD:	Expected Conditional Stochastic Dominance.
GISP:	Global Invasive Species Program.
IP:	Integer Programming.
KP:	Knapsack Problem.
LP:	Linear Programming.
MIP:	Mixed-Integer Programming.
MKP:	Multi-Dimensional Knapsack Problem.

M-SMIP:	Multi-Stage Stochastic Mixed-Integer Programming.
MSS-MIP:	Multi-Stage Stochastic Mixed-Integer Programming.
NISC:	National Invasive Species Council.
NVM:	No Vaccination Model
PPO:	Proximal Policy Optimization.
RA-MSS-MIP:	Risk-Averse Multi- Stage Stochastic Mixed-Integer Programming.
RAM:	Risk-Averse Measure.
RVM:	Random Vaccination Model
SP:	Stochastic Programming.
TSP:	Traveling Salesman Problem.
UBQO:	Unconstrained Binary Quadratic Problem.
USDA:	United States Department of Agriculture.
VaR:	Value at Risk.
WHO:	World Health Organization

CHAPTER 1

INTRODUCTION

Epidemics and more widespread biological infections called pandemics are situations catching even developed societies unprepared which was proven by the latest COVID-19 outbreak. Over the last year, COVID-19 has caused 186,033,321 infections and killed 4,020,869 people worldwide (as of July 8, 2021) [University, 2021], and has paralyzed the world economy causing supply chain disruptions of different industries and an increasing unemployment rate in every country. But it is not just humans; epidemics hit trees as well. Over the past century, uncontrollable pests and fungi have taken out the chestnut blight [Griffin, 2000], the hemlock [Morin and Liebhold, 2015], and now ash trees [Li et al., 2019]. Although the COVID-19 is an invasion caused by the SARS-CoV-2, a coronavirus, it has some differences compared to the other invasive species in the environment.

Invasive species are plant, animal, or pest species that are non-native to a location and have the tendency to overspread and cause possible damage to the environment, human health, and economy [Ehrenfeld, 2010]. Common and Stagl [2005] estimate a total economic cost of invasive species to exceed $120 billion over 85 years only in the US. Even this high cost is actually an underestimate because other losses due to the invasive pests (such as impact in human health, carriers or causes of different diseases, loss of biodiversity) are impossible to be quantified in monetary terms [Pejchar and Mooney, 2009]. These adverse effects of invasive species on different ecosystems have been addressed extensively (see, e.g., Koenig et al. [2013], Paini et al. [2016], Gallardo et al. [2016], DeSantis et al. [2013]).

Managing invasive species involves different tasks, from locating where these species are to dealing with the treatment methodologies. Once an invader usurps a new ecosystem a decision-maker has to come up with a surveillance strategy to locate the pest species. By increasing surveillance efforts, managers have a higher chance to locate the invaded areas and apply early preventive measures to manage the outbreak. The utmost purpose of the manager is to eradicate the invasive species. However, surveillance is costly, and spending more to survey in many cases means less budget to manage the already invaded and discovered locations [Mehta et al., 2007]. Due to limited resources, a more realistic approach is to slow down the spread so that the native species thrive.

Emerald Ash Borer (EAB) is among the most well-known invasive species in the last decade, invading large areas in North America. The EAB is a wood-boring pest native to Asia which was discovered in the U.S. in 2002. Effective management is a critical issue as EAB kills 99% of the ash trees it infests, thus, causing homeowners and governments billions of dollars. The majority of this economic damage from EAB is done in cities and populated areas where ash trees are located in the streets or parks having a high value to the environment [Poland and McCullough, 2006]. Kovacs et al. [2010] estimate the total cost of EAB to the U.S. for the last decade to be $10.7 billion, while McKenney et al. [2012b] estimate a cost of EAB to Canada to be $524 million over a 30-year period. Since its discovery, entomologists have worked hard to develop management strategies to keep ash trees healthy. A Herms and Mccullough [2013] suggest a set of activities to manage the EAB infestation and reduce ash mortality. They propose surveillance at the early stages of the infestation, removal of infested ash trees to slow down the spread, and insecticide treatment to protect high-value trees. Other studies also discuss bio-synthesis or insecticide treatment (see, e.g., Mercader et al. [2015], Jennings et al. [2016], Mercader et al. [2016], Duan et al. [2017]).

While invasive species management differs from human epidemics in that they are specific to a tree type, and no epidemic can hit every tree at once, one can think of the COVID-19 as an invasion caused by the SARS-CoV-2, a coronavirus. Hence, we can build similarities that can be helpful to deal with both of these threats.

In this dissertation, we aim to develop new spatio-temporal modeling and solution approaches using multi-stage stochastic mixed-integer programming and reinforcement learning to represent complex biological invasions of different species and ecosystems by considering a limited budget and considering multiple optimization criteria. Difficulties of the aforementioned models include high computational complexity, spatial and temporal dimensions, definite economic capacities, and representation of different biological properties. Therefore, one main goal of this dissertation is to involve the computing power of machine learning techniques to tackle the computational complexity of these problem formulations.

Our objective is to study the transition mechanism of the biological invasions and the possible management options to formulate strategies that serve the decision-makers to optimally use the budget at hand and save as many healthy ash trees and human lives as possible. Invasive species in general, including EAB, spread over space and time. Thus, we consider large areas infested with EAB and handle the spatial dimension by separating landscapes in proportional areas that we refer to as sites, where we keep track of the tree population in each site, including the trees we know to belong to a certain infestation level. In addition, the inner and outer movements of infection from sites are also modeled on the dispersal mechanism based on the biological properties of the pest.

Different than many plant or insect invasive species, the coronavirus itself does not move, fly, or travel; instead is passed from person to person and moves wherever infected individuals go. Due to this, to model the pest dynamics, we need to model human behavior, which is way more complex than modeling other invasive species in nature.

Despite the power of the optimization-based models, aiming to address an epidemic problem on a large population exhausts the model. Thus, it would be hard to obtain results in a reasonable time. Hence, we investigate the power of machine learning techniques, deep reinforcement learning in particular, on an NP-Hard problem, such as multi-dimensional knapsack problem. To address the complexity of such combinatorial optimization problem, we first formulate a one-dimensional, then a two-dimensional learning environment where decision variables are mapped into cells. A deep reinforcement learning agent is responsible for wandering around in this two-dimensional environment and learning to select items for finding the best solution to large instances of the multi-dimensional knapsack problem.

Due to its direct threat to human life, COVID-19 very fast became a pressing matter for the governments. Prevention measures involved physical distancing, wearing masks, hand hygiene, and quarantining when not feeling well. At first, medical researchers needed time to investigate the transmission ways, the incubation period for newly infected individuals, and other biological properties. Having this information helps us model the spread and simulate invasions more accurately.

Observing the power of deep learning, we extend our deep reinforcement learning approach to address decision-making over a large population. We extend an agent-based simulation [Kerr et al., 2020] that models person-to-person contact for different communities in large populations, to allow for real-time management interventions from a governing party, a deep reinforcement learning agent. In addition, we extend to a compartmental model flexible to one-shot and two-shot vaccination. Using such a framework, we are able to mimic a government-population situation, where one imposes restrictions, and the latter obeys them. Our results have shown that the trained deep reinforcement learning agents can build knowledge regarding possible optimal interventions when faced with different states of the epidemic.

1.1 Background in Methodology

Researchers have utilized optimization-based methods to assist the cost-effective resource allocation, also considering the benefits of taking action comparing costs of damage from invasions against that of detecting the invaders and managing the outbreak [Hof, 1998, Mehta et al., 2007, Büyüktahtakın et al., 2011, Epanchin-Niell et al., 2012, Kovacs et al., 2014, Büyüktahtakın et al., 2015, Büyüktahtakın and Haight, 2018, Quick et al., 2017, Büyüktahtakın et al., 2014a, Büyüktahtakin et al., 2014]. Some other studies attempt to model pest surveillance and control under the assumption of uncertain spread [Horie et al., 2013, Yemshanov et al., 2017]. These models considered a time domain with only two periods. Onal et al. [2020] addressed the optimal search path for locating and controlling the infestation of a biological invader using a multi-period simulation-optimization framework. Kıbış et al. [2021] addressed the problem of joint optimization of surveillance and control decisions with a multi-stage stochastic mixed-integer programming (MSS-MIP) formulation that extended the time horizon to five stages and applied that model to the management of EAB in Burnsville, Minnesota, USA. We believe that to address an invasive species problem, multi-stage stochastic programming is the right tool as it considers the problem long-term and accounts for the uncertainty that comes with the biological invader. MSS-MIP combines the complexity of stochastic programming with a mixed-integer programming model and represents an NP-hard combinatorial problem. Recent developments of multi-stage MIP solving techniques have been limited [Birge and Louveaux, 2011]. Multi-stage stochastic programming has been widely used in many fields, including but not limited to healthcare [Yin and Büyüktahtakın, 2020], forestry [Kıbış et al., 2021], agriculture [Cobuloglu and Büyüktahtakın, 2017] and finance [Abdelaziz et al., 2007]. Typically, multi-stage programs maximize (minimize) an expectation criterion over all possible scenarios, each of which has a certain probability of occurrence. The expectation criterion is very useful in situations where there is no indication that an extreme event can be observed.

Traditional two-stage and multi-stage stochastic programs consider only an expectation criterion in the objective function of the optimization problem based on the probability of each scenario, also known as a risk-neutral approach. In problems containing outliers in the distribution of the scenarios, the risk-neutral approach may perform poorly. Assuming that the manager wants to be careful of these extreme scenarios, in the risk-neutral approach, these undesirable outcomes associated with an adverse scenario cannot be prevented. The solution obtained from optimizing the expected objective function will perform poorly when one of these outlier scenarios happens. In such cases, risk-averse models become necessary. For example, in a disaster management situation [Escudero et al., 2018a], non-repetitive decisions, such as facility locations [Escudero et al., 2017], may result in a substantial operational cost or even an inability to fulfill the demand for a specific realization of the random parameters [Escudero et al., 2018b].

Decomposition algorithms are the mainstream methods to tackle two-stage stochastic MIPs. The non-convex region formed in multi-stage MIP problems cannot be tackled using direct decomposition. Most solution approaches are based on stage-wise (resource-directive) or scenario-based (price-directive) decomposition. Recent studies in the last decades integrated solution approaches of Stochastic Programming (SP) and discrete optimization methods (IP). For example, Bender's decomposition is applicable to a class of two-stage stochastic problems where the first-stage decisions are mixed-integer, and the recourse decisions are found by solving the linear programming (LP) models [Sen, 2005]. Most studies of MSS-MIP problems decomposed them into multiple scenarios and treated each scenario as a separate problem. For example, the study of CarøE and Schultz [1999] treated the solution of the Lagrangian dual as a lower bound of the original problem by relaxing the non-anticipative constraints. Heuristic algorithms were used to provide an upper bound on the dual solution, and branch and bound was used to find a feasible integer solution. Scenario-based decomposition approaches, such as Lagrangian and Dantzig-Wolfe, have been shown to be effective in different multi-stage stochastic integer problems [Nowak and Römisch, 2000, Lulli and Sen, 2004].

The majority of solution approaches presented for risk-averse multi-stage stochastic optimization problems are extensions of the solution techniques proposed for the risk-neutral equivalents Birge and Louveaux [2011]. For example, Shapiro et al. [2013] and Philpott and De Matos [2012] have extended the stochastic dual dynamic programming algorithm to risk-averse problems. Schultz and Tiedemann [2006] develop a Lagrangian decomposition algorithm to solve the scenario-based formulation of two-stage mixed-integer stochastic programming involving CVaR. Zhang et al. [2016] use a nested L-shaped method and investigated multiple cuts to improve the efficiency of a risk-averse multi-stage program. Guo and Ryan [2017] obtain lower bounds using the progressive hedging algorithm to solve time-consistent risk-averse multi-stage stochastic integer programs.

At first, we model a multi-stage stochastic programming model of EAB surveillance and control decisions [Bushaj et al., 2021b, Kıbış et al., 2021] only considering an expectation criterion in the objective function, the popular Risk Neutral measure. However, due to the intrinsic biological characteristics of the invader and some outside factors, such as careless transportation of infested wood, an infestation could spread fast, and substantial losses of ash trees could happen in a shorter time frame than expected. To alleviate the adverse impacts of experiencing such events, we consider a risk measure in the objective function in addition to the expectation criterion. The incorporation of the risk factors complicates the model, thus requiring advanced computational methodologies to solve it. To tackle the computational difficulty of the proposed complex risk-averse multi-stage stochastic mixed-integer program, we implement the scenario dominance cutting plane algorithm introduced in Büyüktahtakın [2020] to solve the risk-averse multi-stage stochastic mixed-integer programming (RA-MSS-MIP) model more efficiently. The effectiveness of these cuts is studied under the risk-neutral and risk-averse models. We provide insights on how risk-aversion affects decision-making, such as the budget allocated to insecticide treatment and tree removal. We also analyze the benefits of ash trees under risk compared to the original expectation criterion in the risk-neutral problem.

Mathematical programming models are powerful solution methods but mostly suffer from the curse of dimensionality. As the size of the problem instances increases, it becomes quite challenging to get a good solution in a considerable amount of time. Hence, recent literature shows a synergy between operations research problems and the usage of powerful machine learning algorithms [Oroojlooyjadid et al., 2020, 2017, Nazari et al., 2018]. The Multi-Dimensional Knapsack Problem (MKP) is a knapsack problem with multiple constraints. The MKP lies at the core of many combinatorial optimization problems with applications varying from agriculture [Kantas et al., 2015, Cobuloglu and Büyüktahtakın, 2014, 2015a], production planning [Büyüktahtakın et al., 2018b, Büyüktahtakın and Liu, 2016, Hartman et al., 2010], and asset management [des Bordes and Büyüktahtakın, 2017, Büyüktahtakın et al., 2014b, Büyüktahtakın and Hartman, 2016]. MKP can also be considered as a special case of integer programming, except it restricts the decision variables to 0 or 1. The difficulty of MKP lies in the fact that optimal solutions cannot be reasonably obtained as the instances grow. MKP can be easily treated as a sequential decision-making problem, making solution methods important in a wide range of applications, such as business, logistics, computer networks, and invasive species management.

Despite many research efforts worldwide and multiple solution approaches to solve the MKP, there is still a large room for improvement, especially for large-scale instances. Considering the recent empowerment of machine learning methods for tackling optimization problems presents a wide area to explore. In particular, reinforcement learning is a promising candidate to outperform current approaches for large MKP instances. Literature suggests that reinforcement and deep reinforcement learning (DRL) approaches can learn solution strategies to solve combinatorial optimization problems [Ma et al., 2019, Bello et al., 2016, Barrett et al., 2020]. Inspired by this, we investigate a reinforcement learning approach to help solve large instances of a core MIP problem. We develop a framework that aims to generalize solving MKP instances of different sizes and distributions.

1.2 Motivation and Contributions

Invasive species threaten the health of an environment the same way a pandemic threatens the health of a human population. We believe similar solutions can be designed to solve these problems. Hence, in this dissertation, we first expand the state-of-the-art methodology with cutting plane methods and mathematical solutions and develop a novel reinforcement learning framework to tackle the core resource allocation problem. Then we extend these problem formulations and the RL framework to agent-based simulation as an accurate process to mimic real-world situations and optimize decision strategies developing machine learning-based techniques.

First, we formulate a multi-stage stochastic mixed-integer programming (MSS-MIP) problem that optimizes surveillance and management decisions for controlling a pest outbreak in urban environments. To this formulation, we also contribute a method to compute the time-dependent probabilities based on surveillance history. In addition, a distance-dependent infection spread is used to model the movement dynamics of EAB. With the interest of the experts and forest agencies, we extend our MSS-MIP formulation to a risk-averse multi-stage stochastic mixed-integer programming formulation (RA-MSS-MIP), which to our knowledge, is the first

RA-MSS-MIP formulation for invasive species literature. To tackle the difficulty of this high-dimensional problem, we adapt the scenario dominance cutting planes introduced by Büyüktahtakın [2020] to the case of decision-dependent uncertainty, resulting in a considerable improvement in solution time of those NP-Hard problem instances. Based on our research, we recommended the following actions to resource managers:

> Our approach demonstrates that timely detection and early response are critical factors for maximizing the number of healthy trees in urban areas affected by pest outbreaks.
>
> Treatment of the infested trees is most effective when done at the earliest stage of infestation. Treating asymptomatic trees at the earliest stages of infestation provides higher net benefits than tree removal or no-treatment options.
>
> Our analysis suggests the use of branch sampling as a more accurate method than the use of sticky traps to detect the infested asymptomatic trees, which enables treating and removing more infested trees at the early stages of infestation.
>
> Our results also emphasize the importance of allocating a sufficient budget for tree removal to manage EAB infestations in urban environments. Tree removal becomes a less useful option in small-budget solutions where the optimal policy is to spend most of the budget on treatments.
>
> Our results suggest that, as the manager becomes more risk-averse, insecticide treatment becomes less preferred compared with tree removal, especially for scenarios that involve high infestation spread each year. This is an important practical finding because forest managers often debate over two broad strategies for EAB management: surveillance and insecticide treatment of ash trees versus surveillance and staged removal of ash trees. Our results suggest that the former strategy may provide a higher expected net benefit while the latter strategy may provide a lower risk of outcomes with very low net benefits. The selection of the appropriate strategy will depend on the risk preference of the decision-maker.
>
> Based on our results, increasing risk-aversion by emphasizing the poorly performing scenarios in the objective function might come at a price of reduced expected net benefit. Despite this price, the manager may see this loss as a worthy sacrifice towards the mitigation of possible disaster scenarios.
>
> Our analysis of surveillance frequency suggests that as we survey less, the expected risk increases due to higher uncertainty about infestation realizations. Thus, a risk-averse manager would want to survey more often.

The contributions of this dissertation extend to combinatorial optimization problems as well. Developing a novel DRL framework, we tackle the computational difficulty of the multi-dimensional knapsack problem. This framework can solve instances of different sizes and distributions faster than the state-of-the-art, CPLEX [Cplex, 2009], with only an additional gap of about 0.2%, which is a quite significant achievement. Utilizing the power of deep learning, we then extend our DRL framework to epidemic control planning. For that, we integrate an agent-based simulation model with our DRL framework. For our simulation, we extend an agent-based model presented in Kerr et al. [2020]. Kerr et al. [2020] provide a realistic agent-based model involving interventions and different contact layers for each individual in the population. We extend the health compartments to include vaccination and also make the model fit to update decisions in a set frequency by the decision-maker. As a central decision-maker, a state-of-the-art Deep Q-Network (DQN) agent is used. The agent is trained on the compartmental data provided by the agent-based simulation. We train DRL agents on different periods of COVID-19 and draw conclusions on how the situation was handled by government agencies and what could have been a better action. Our results also show that the simulation-reinforcement learning framework can suggest meaningful actions provided various states of the epidemic.

The research contributions of this dissertation are discussed in detail in each chapter, respectively.

1.3 Summary of Research Objectives and Accomplishments

The goal of this dissertation is to provide new methodological contributions to solving resource allocation problems by presenting new multi-stage stochastic programming and deep reinforcement learning methods with an application to the control of invasive species and epidemic diseases. Below, we summarize the research objectives and accomplishments achieved under each chapter of this dissertation.

Chapter 2 provides an introduction to the invasive species management problem and methodologies used. Emerald Ash Borer (EAB), a wood-boring insect native to Asia, was discovered near Detroit in 2002 and has spread and killed millions of ash trees throughout the eastern United States and Canada. EAB causes severe damage in urban areas where it kills high-value ash trees that shade streets, homes, and parks and costs homeowners and local governments millions of dollars for treatment, removal, and replacement of infested trees. We present a multi-stage, stochastic, mixed-integer programming model to help decision-makers maximize the public benefits of preserving healthy ash trees in an urban environment. The model allocated resources to surveillance of the ash population and subsequent treatment and removal of infested trees over time. We explore the multi-stage dynamics of an EAB outbreak with a dispersal mechanism and apply the optimization model to explore surveillance, treatment, and removal options to manage an EAB outbreak in Winnipeg, a city of Manitoba, Canada.

In Chapter 3, we derive a nested risk measure for a maximization problem and implement it in a scenario-based formulation of a multi-stage stochastic mixed-integer programming problem. Specifically, we extend the multi-stage, stochastic, mixed-integer programming formulation developed in Chapter 2 to a risk-averse formulation for the optimal surveillance and control of the EAB. We present a mean-Conditional Value-at-Risk (CVaR), multi-stage, stochastic mixed-integer programming model to optimize a manager's decisions about surveillance and control of EAB. The objective is to maximize the benefits of healthy ash trees in forests and urban environments under a fixed budget. Combining the risk-neutral objective with a risk measure allows for a trade-off between the weighted expected benefits from ash trees and the expected risks associated with experiencing extremely damaging scenarios. We solve the model using the Scenario Dominance Cuts (SDC) algorithm under different risk parameters. We define scenario dominance cuts for the maximization problem and test their performance relative to the CPLEX solution. We apply our risk-averse model to the case of EAB management in the state of New Jersey. Computational results demonstrate that our CVaR risk-averse approach improves the least-benefit scenarios compared to the risk-neutral model. Our results show a shift in the optimal strategy from applying less expensive insecticide treatment to more costly tree removal as the manager becomes more risk-averse. We also find that risk-averse managers survey more often to reduce the risk of experiencing adverse outcomes.

The problems discussed in Chapters 2 and 3 of this dissertation are large NP-Hard combinatorial optimization problems. In Chapter 4, we address the difficulty of solving large-scale combinatorial optimization problems presenting a novel deep reinforcement learning (DRL) framework. Initially, we propose an unsupervised learning approach using K-means to obtain a reasonable initial feasible solution that is used to train the DRL agent. We also propose a heuristic to reduce the dimension of Multidimensional Knapsack Problem (MKP) instances in our DRL framework. We estimate a worthiness value for each item and re-order them by considering all affecting factors in the MKP formulation. Using K-means and the item worthiness ratio obtained from the heuristic as facilitators, we build a 2-D RL environment to represent MKP instances of different sizes. We propose a Deep Reinforcement Learning (DRL) framework where we aim to train different agents compatible with a discrete action space for sequential decision-making while still satisfying any resource constraint of the MKP. This novel framework incorporates the decision variable values in the 2-D DRL where the agent is responsible for assigning a value of 1 or 0 to each of the variables. To our knowledge, this is the first DRL model of its kind where a 2D environment is formulated, and an element of the DRL solution matrix represents an item of the MKP. Our framework is configured to solve MKP instances of different dimensions and distributions. We train four different agents in our framework and present results by comparing each of them to the CPLEX commercial solver. Results show that our agents can learn and generalize over instances with different sizes and distributions. Our DRL framework shows that it can solve medium-sized instances at least 45 times faster in CPU solution time and at least 10 times faster for large instances with a maximum solution gap of 0.28% compared to the performance of CPLEX. Furthermore, at least 95% of the items are predicted in accordance with the CPLEX solution.

In Chapter 5, we take advantage of the solution approach developed in Chapter 4 and an agent-based simulation model to address the controversies of epidemic control planning by using a novel Simulation-Deep Reinforcement Learning (SiRL) framework. COVID-19 reminded constituents over the world that government decision-making could change their lives. During the COVID-19 pandemic, governments were concerned with reducing the fatalities of the virus spread but at the same time also maintain a flowing economy. We aim to provide insights into epidemic decision-making regarding what interventions are necessary at a point in time of the epidemic based on the purpose of the decision-maker. Further, we intend to compare different vaccination strategies to shine a light on who should get priority in the vaccination process. To address these issues, we propose a simulation-deep reinforcement learning framework. This framework is composed of an agent-based simulation model and a governor DRL agent that can enforce interventions in the agent-based simulation environment. Our results show that our framework can help suggest optimal actions according to a specific epidemic situation. Our DRL agent can learn effective strategies based on a reward structure. We compare our DRL decisions towards the government interventions at different periods of time during the COVID-19 pandemic. In addition, we compare the results of an age-based and random vaccination strategy. Our results suggest that the timely introduction of interventions could have reduced the damage done by the COVID-19 pandemic. Furthermore, if a random vaccination would have been used, infections would reduce by 32% and a better economic situation would be maintained.

1.4 Organization of the Dissertation

This Ph.D. dissertation is organized in chapters that correspond to four journal papers. Chapter 2 addresses the development of a multi-stage stochastic mixed-integer program and its application to an urban environment in Manitoba, Canada. Chapter 3 extends the MSS-MIP to a risk-averse MSS-MIP and implements scenario dominance cutting planes to the decision-dependent scenario formulation. Chapter 4 proposes a deep reinforcement learning approach to solve multi-dimensional knapsack problem instances. Chapter 5 presents a framework composed of an agent-based simulation and deep reinforcement learning developed for epidemic control planning. Finally, in Chapter 6, we summarize our contributions and further research directions inspired by this dissertation.

CHAPTER 2

OPTIMIZING SURVEILLANCE AND MANAGEMENT OF EMERALD ASH BORER IN URBAN ENVIRONMENTS

2.1 Introduction

Invasive species are plant, animal, or pest species that are non-native to a location and have the tendency to overspread and cause possible damage to the environment, human health, and economy [Ehrenfeld, 2010]. The harmful effects of invasive species on agriculture, aquatic, forests, and ecosystems have been studied extensively [Koenig et al., 2013, Paini et al., 2016, Gallardo et al., 2016, DeSantis et al., 2013]. The estimated total economic cost from invasive species in the U.S. alone was expected to exceed $120 billion over 85 years [Pimentel et al., 2005]. The extent of the damage may be an underestimate because many losses caused by invasive species (such as loss of biodiversity, indirect impacts on human health, and loss of ecosystem services) are difficult to estimate in monetary terms [Pejchar and Mooney, 2009].

Management of invasive pests requires substantial resources to locate and control the established pest populations. Once the invader is established in a novel ecosystem, two sets of decisions need to be made on how to survey the area and how to manage the outbreak. By increasing surveillance to detect invasive species, managers may increase their chances of finding a species early at lower population sizes, lessening the extent of damages, and making subsequent control potentially less expensive and more effective. However, detecting invasive species requires costly surveillance, which limits the manager's options to control the infestation when budgets are limited [Mehta et al., 2007]. Optimization-based methods have been widely used to assist with cost-effective resource allocation and address the trade-offs between incurring the costs of damage from invasions versus the costs to detect and manage the outbreaks [Hof, 1998, Mehta et al., 2007, Büyüktahtakın et al., 2011, Epanchin-Niell et al., 2012, Kovacs et al., 2014, Büyüktahtakın et al., 2015, Büyüktahtakın and Haight, 2018].

In this work, we present a linear integer programming model that optimizes surveillance and management decisions for controlling a pest outbreak in an urban environment. The model is formulated as a multi-stage stochastic mixed-integer programming problem (M-SMIP) based on the work of Kıbış et al. [2021]. We applied the model to assess the options to manage the infestation of emerald ash borer (EAB) in Winnipeg, a city of Manitoba, Canada. The insect poses a major threat to North American ash species [Haack and Petrice, 2003, A Herms and Mccullough, 2013] and has already caused major damage to urban and natural forests in the eastern and central U.S. and Canada [Kovacs et al., 2010, 2014, McKenney et al., 2012a]. Long-distance EAB spread has been associated with human activities, primarily with commercial and passenger vehicles that could potentially move firewood or other infested materials [Haack, 2006, Haack et al., 2010, Kovacs et al., 2010, Koch et al., 2011, Yemshanov et al., 2015]. There is also evidence that the pest can hitchhike on vehicles [Buck and Marshall, 2008]. Timely detection of new EAB infestations is difficult because insect damage is not immediately visible, and as a result, new detections usually indicate the presence of large, established populations that are difficult to control [McCullough et al., 2009, Ryall et al., 2011].

Major economic damage from EAB infestations occurs in cities and populated places where high-value ash trees grow along streets or in parks [Poland and McCullough, 2006]. The total cost of the EAB outbreak to property owners and local governments is estimated to be $10.7 billion in the last decade in the U.S. [Kovacs et al., 2010] and $524 million over a 30-year period in Canada [McKenney et al., 2012b]. Since its discovery, much work has been done to develop management strategies that reduce ash mortality from EAB infestations. A Herms and Mccullough [2013] proposed various management activities, including surveillance to locate EAB populations at early stages of infestation, treatment of trees with insecticide to protect high-value trees and reduce larval populations, and removal of infested ash trees to slow EAB spread.

Our objective is to determine the optimal location and intensity of ash surveillance, treatment, and removal each year (period) over a five-year horizon. We consider a budget-constrained problem that maximizes the total benefits of maintaining healthy ash trees in an urban area over a 5-year period minus the penalty associated with the presence of the infested trees. We separate the landscape into 1-km^2 management units (sites). For each unit, we know the number of ash trees and build a scenario tree that depicts a set of possible surveillance and management decisions and accounts for the uncertainty associated with EAB population growth and spread. The scenario tree includes two decisions in each period. First decision is whether or not surveillance is applied, and second decision is on the intensity of treatment and removal of the infested trees, depending on the outcome of surveillance. The uncertainty about the tree status after the surveillance is depicted with a set of possible infestation outcomes and associated probabilities before a management decision is made. At the beginning of the planning horizon, we describe the ash population in each unit by the number of ash trees at different stages of infestation based on prior knowledge about the infestation. The model then projects changes in the ash population each year based on assumptions about EAB spread within and between units. If surveillance is undertaken, the model estimates the number of infested trees and makes decisions to remove or treat the infested ash trees, and then updates the probabilities associated with the expected levels of infestation for the next period.

2.1.1 Literature Review

Optimization models have been widely used to assist with surveillance and management of invasive species populations [Mehta et al., 2007, Baxter and Possingham, 2011, Epanchin-Niell et al., 2012]. Several studies considered pest control measures that include the removal of infested or susceptible host organisms and applying chemical or biological control treatments to eradicate or slow the invasion [Hof, 1998, Büyüktahtakın et al., 2011, Büyüktahtakın and Haight, 2018]. Some studies addressed pest surveillance and control under the assumption of uncertain spread [Horie et al., 2013, Yemshanov et al., 2017]. These models considered a time domain with only two periods. Onal et al. [2020] present a multi-period simulation-optimization framework to find the optimal search path for locating and controlling the infestation of a biological invader. Kıbış et al. [2021] addressed the problem of joint optimization of surveillance and control decisions with a multi-stage stochastic mixed-integer programming (MSS-MIP) formulation that extended the time horizon to five stages and applied that model to the management of EAB in Burnsville, Minnesota, USA. MSS-MIP combines the complexity of stochastic programming with a mixed integer programming model and represents an NP-hard combinatorial problem. Recent developments of multi-stage MIP solving techniques have been limited [Birge and Louveaux, 2011]. Decomposition algorithms are the mainstream methods to tackle two-stage stochastic MIPs. The non-convex region formed in multi-stage MIP problems cannot be tackled using direct decomposition. Most solution approaches are based on stage-wise (resource-directive) or scenario-based (price-directive) decomposition. Recent studies in the last decades integrated solution approaches of Stochastic Programming (SP) with discrete optimization methods (IP). For example, Bender's decomposition is applicable to a class of two-stage stochastic problems where the first-stage decisions are mixed-integer and the recourse decisions are found by solving the linear programming (LP) models [Sen, 2005]. Most studies of

MSS-MIP problems decomposed them into multiple scenarios and treat each scenario as a separate problem. For example, the study of CarøE and Schultz [1999] treated the solution of the Lagrangian dual as a lower bound of the original problem by relaxing the non-anticipative constraints. Heuristic algorithms were used to provide an upper bound on the dual solution, and branch and bound was used to find a feasible integer solution. Scenario-based decomposition approaches, such as Lagrangian and Dantzig-Wolfe have been shown to be effective in different multi-stage stochastic integer problems [Nowak and Römisch, 2000, Lulli and Sen, 2004].

In our multi-stage model, the number of variables, constraints, and scenarios increases exponentially for each additional time period. Therefore, in order to improve on the solution time and memory, we apply cutting planes and a pre-processing algorithm adapted from Kıbış et al. [2021]. These cutting planes helped improve the solution time by a factor of 10+ compared to the standard approach without using cutting planes.

2.1.2 Key Contributions of the Chapter

We extend the MSS-MIP formulation of Kıbış et al. [2021] to a study area covering 472 km^2 in the city of Winnipeg, Manitoba, Canada. Our model accounts for the uncertainty about the infestation levels at each site, which is partially resolved by the surveillance decisions.

Our model differs from the study of Kıbış et al. [2021] in the following ways. We depict the temporal dynamics of the infested trees with four infestation levels compared to a five-level model in Kıbış et al. [2021]. The first level represents healthy trees, the second level includes asymptomatic infested trees, the third level includes infested trees with visible signs of infestation, and the fourth level represents dead trees.

For each year of the planning horizon, we compute realization probabilities of possible ash infestation outcomes that may be observed via surveillance. Compared to the formulation in Kıbış et al. [2021] that used constant realization probabilities, our probabilities are time-dependent. We present a new heuristic algorithm that dynamically updates the probabilities of the uncertain infestation outcomes based on the outcomes of the previous survey.

To account for EAB spread we apply a distance-dependent estimation of spread probabilities at four 1-km distance classes from the infested sites, as opposed to the 1-level spread considered in Kıbış et al. [2021]. This spread model captures the short-range spread of EAB in urban environments, as suggested by records from previous EAB surveys in Twin Cities, Minnesota [Osthus, 2017].

Based on the experiences from previous EAB survey campaigns, we assume that only a fraction of the trees can be inspected at a site due to cost and personnel constraints. To account for the incomplete survey, we used a surveillance efficiency parameter to depict the percentage of infested trees detected after inspecting a proportion of host trees. The new surveillance-efficiency parameter introduced in our study compensates for the uncertainty of the surveillance outcomes.

We also compared two common EAB survey methods: applying branch sampling with debarking to detect EAB galleries and placing sticky traps baited with EAB pheromone and ash volatiles to capture EAB adults. We explore the trade-off between the surveillance efficiency and its cost for each survey method over a 5-year planning period.

2.2 Optimization Model

In this section, we present a revised version of the multi-stage stochastic mixed-integer formulation originally proposed by Kıbış et al. [2021] for the optimal surveillance and control of EAB. This modified formulation improves over the former one by addressing important issues regarding the surveillance uncertainty as well as the dynamic probabilities of uncertain surveillance outcomes.

2.2.1 Problem Definition

We formulate the objective of the EAB surveillance and control problem as maximizing the number of uninfested, healthy ash trees in a landscape at the end of the planning period subject to an upper bound on the budget for surveillance and control. Delimiting surveys typically divide the area of concern into a grid of survey sites where a sample of ash trees is inspected at each site. The ash population in each site is divided into healthy trees that are susceptible to infestation and infested trees belonging to three classes (levels): asymptomatic trees, symptomatic trees, and dead trees. Infested trees are the source of EAB spread to susceptible trees in surrounding sites. Each year, infested trees transition to the next, more severe infestation level, and susceptible trees may become infested through EAB spread within and between neighboring sites. Asymptomatic trees represent the lowest infestation level 1, followed by symptomatic trees with visible signs of infestation (level 2) and, if no treatment or removal is applied, trees die (level 3) Figure 2.1. We assume that decisions to treat the infested trees can be only effective when applied to asymptomatic trees while symptomatic and dead trees may be removed. A time stage in this study refers to a time period in the stochastic programming model. We set each time period to one year because EAB generally has a 1-year life cycle and ash trees may die after 3 to 4 years of heavy infestation, although it is difficult to determine the time of first infestation [McCullough and Katovich, 2004]. The visual ash tree canopy condition assessment data collected by Flower et al. [2013] shows that ash trees progress from one level to the next within one year on average. Recent evidence suggests that EAB may switch to a two-year life cycle in a colder climate

[Cappaert et al., 2005] and in particular in Winnipeg, so our current assumption depicts a pessimistic view of infestation outcomes.

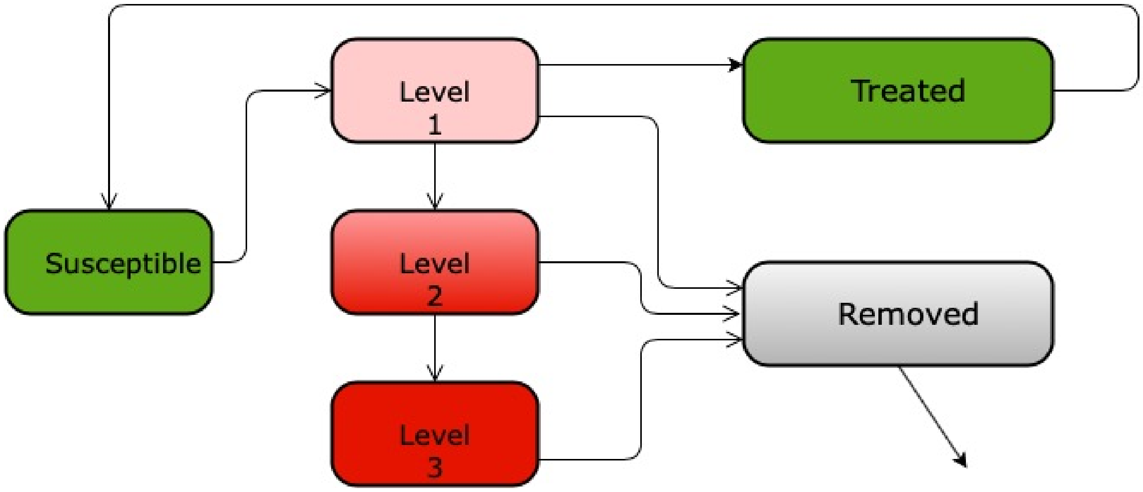


Figure 2.1 Transitions between the infestation levels.

The number of healthy trees that may be infested at a survey site is uncertain, which required using a probabilistic depiction of spread. In our case, the number of newly infested trees at a site is a random variable that depends on the number of EAB adults produced at a given site and neighboring sites. Surveillance decisions are critical because they provide information about the infestation, which allows making decisions about ash treatment and removal. Due to the high cost of surveillance, pest surveys limit inspections to a small sample of trees. A partial observation provides limited knowledge about the actual number and location of the infested trees. Because only a portion of all trees is inspected the actual number of the infested trees after surveillance is unknown, and treatment and removal measures are applied to both infested and healthy trees. We address this uncertainty with a surveillance efficiency parameter that denotes the percentage of infested trees detected after surveillance and serves as a multiplier to adjust the number of detected infested trees that can be treated or removed and update the number of remaining infested trees at a surveyed site.

The progression of tree infestation and an associated sequence of management decisions is shown in Figure 2.2. In the first period, the actual level of infestation at a survey site is unknown, and management decisions can only be made in the next period after the site is surveyed. If asymptomatic trees (infestation level 1) are treated in the second period, they become immune to infestation and regain the susceptible tree status in period three. Symptomatic and dead trees at infestation levels 2 and 3 can only be removed. In the third period, no surveillance is applied, and due to the uncertainty in infestation spread, the numbers of susceptible trees and trees at infestation level 1 are expectations based on the outcomes of the survey in the previous period. In period four, no survey is applied, and we only have partial information from period three. In period five, surveillance is applied, which provides an estimated number of trees at each infestation level and allows making decisions on treatment and removal.

2.2.2 Scenario Tree

Our scenario tree depicts the sequences of possible surveillance decisions and the stochastic infestation outcomes over time and is based on the model of Kıbış et al. [2021]. Full details are provided in Appendix A, here we provide only a general description. The scenario tree starts in period one and at each branching progresses through time based on realizations of the uncertain levels of infestation (see example in Figure 2.3). Two types of nodes in the scenario tree indicate the surveillance (black circles) and no surveillance conditions (empty circles). Square nodes represent treatment or removal decisions. Given the uncertainty about EAB spread, each decision has two possible outcomes: high or low realization of the uncertain level of infestation. The outcome without surveillance (identified by empty circles) yields the expected value of the uncertain infestation level. The notation on each arc, p_t^H , p_t^L, and p_t^M, stands for the probability of detecting the infestation at a high (H) or

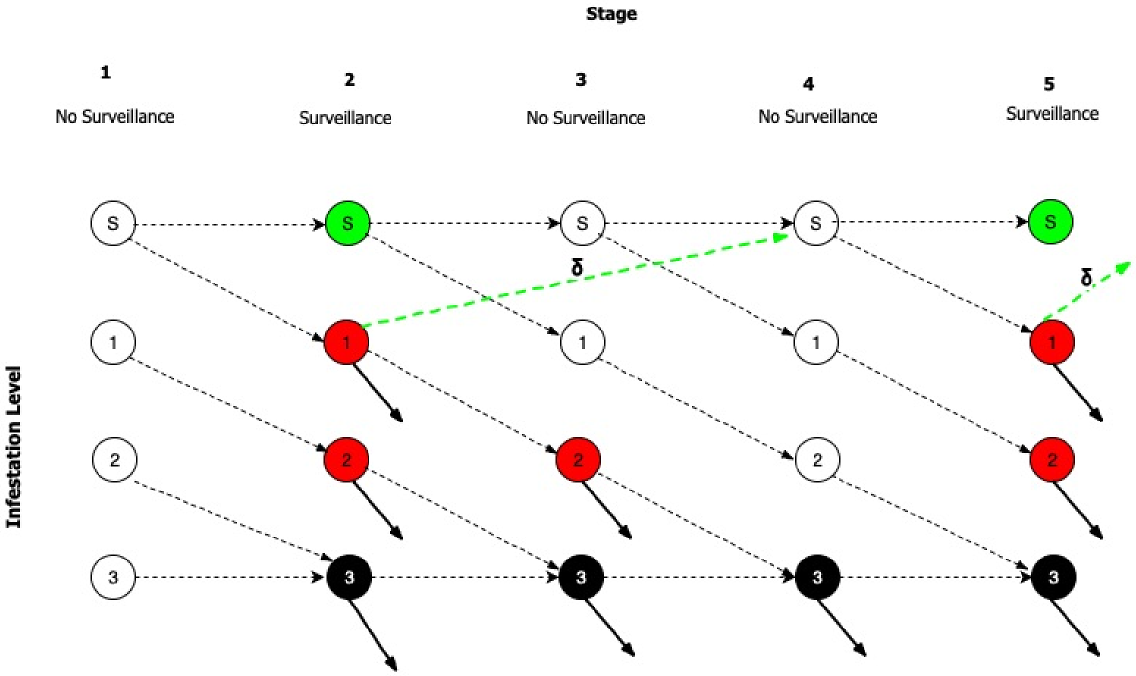


Figure 2.2 An example transition diagram of the ash tree population with over a 5-period planning horizon under a particular surveillance regime (i.e., surveys in years 2 and 5 and no surveillance in years 1, 3 and 4). X-axis denotes time periods and y-axis represents tree infestation levels. Node S denotes the susceptible trees and nodes 1, 2, and 3 denote the infestation levels 1-3 (asymptomatic, symptomatic and dead trees). In each stage, uncolored nodes represent the infestation levels for which the actual number of infested trees is unknown because no surveillance was made. Light shaded (green) nodes depict the detected susceptible trees; dark-shaded (red) nodes depict trees at infestation levels 1-3 after surveillance; and black nodes represent the infested dead trees that must be removed. Dashed green lines show the number of treated trees that become temporarily immune to an infestation and are moved to a pool of susceptible trees; dotted lines show the transition in tree infestation level from one period to another; and bold arrows show infested dead trees that are removed. *Source: Adapted from Kıbış et al. [2021].*

low (L) levels, or the expected infestation level (M) in the absence of surveillance in period t, producing 3_t scenarios. We assume a medium infestation level as an expected outcome without surveillance and use this level to update the number of infested trees at a given site without surveillance. Terms $\pi_1 - \pi_{3t}$ denote the conditional probabilities of a scenario $\omega, \omega \in [1, 2, \ldots, 3^t]$. This probability is calculated by multiplying the probability of each realization through the whole scenario path and normalizing it over all scenarios so the sum of the probabilities over all scenarios is equal to 1. An example of a two-stage scenario tree is depicted in Section A.1.

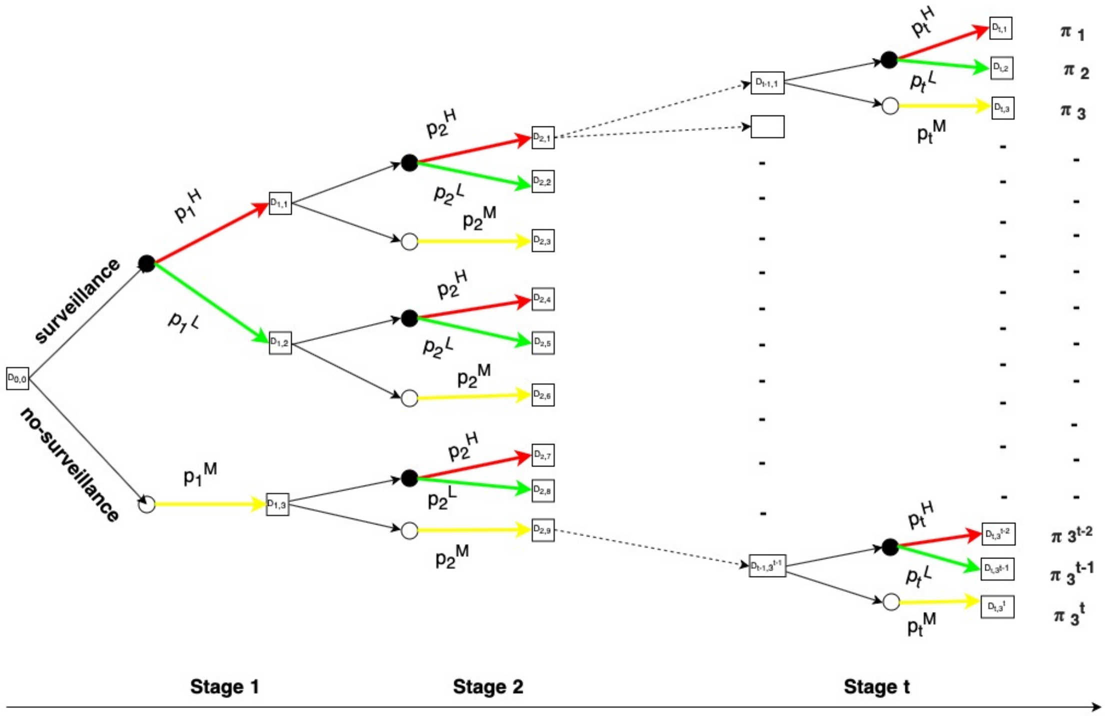


Figure 2.3 Multi-stage scenario tree. Terms p_t^H and p_t^L denote the realization probabilities of high (H) and low (L) levels of infestation after surveillance; p_t^M denotes a default realization of medium infestation level (M) without surveillance. Black circles represent nodes with decisions after the surveillance; white circles depict nodes without surveillance; arcs leaving black and white circles depict possible realizations of the estimated beliefs about the number of susceptible and infested trees; red arrows depict realizations of high infestation levels; green arrows depict realizations of low infestation levels; yellow arrows depict anticipated levels of infestation without surveillance based on the initial belief. Squares $D_{t,s}$ depict treatment and removal decisions.
Source: Adapted from Kıbış et al. [2021].

2.2.3 Algorithm for Scenario Tree Probability

We computed the expected values of the infestation probabilities for each period in the scenario tree using the uncertain outcomes of spread. Given that the surveillance outcomes are unknown, we assumed equal probabilities for realizations of high and low infestation levels in time period one. We then updated the realization probabilities for future periods based on the outcomes of the surveys done in previous periods. For each scenario in a multi-stage planning problem, we have calculated the probabilities of infestation with a new heuristic algorithm. This algorithm assigns higher probabilities to reoccurring events and assumes that realizations of infestation outcomes are time-dependent. For example, if a particular realization is observed in time period t, the likelihood of having the same realization in time period $t+1$ is higher. Section A.2, Algorithm 5, describes the heuristic algorithm and shows a probability calculation example for the three-stage scenario tree depicted in Figure 2.4.

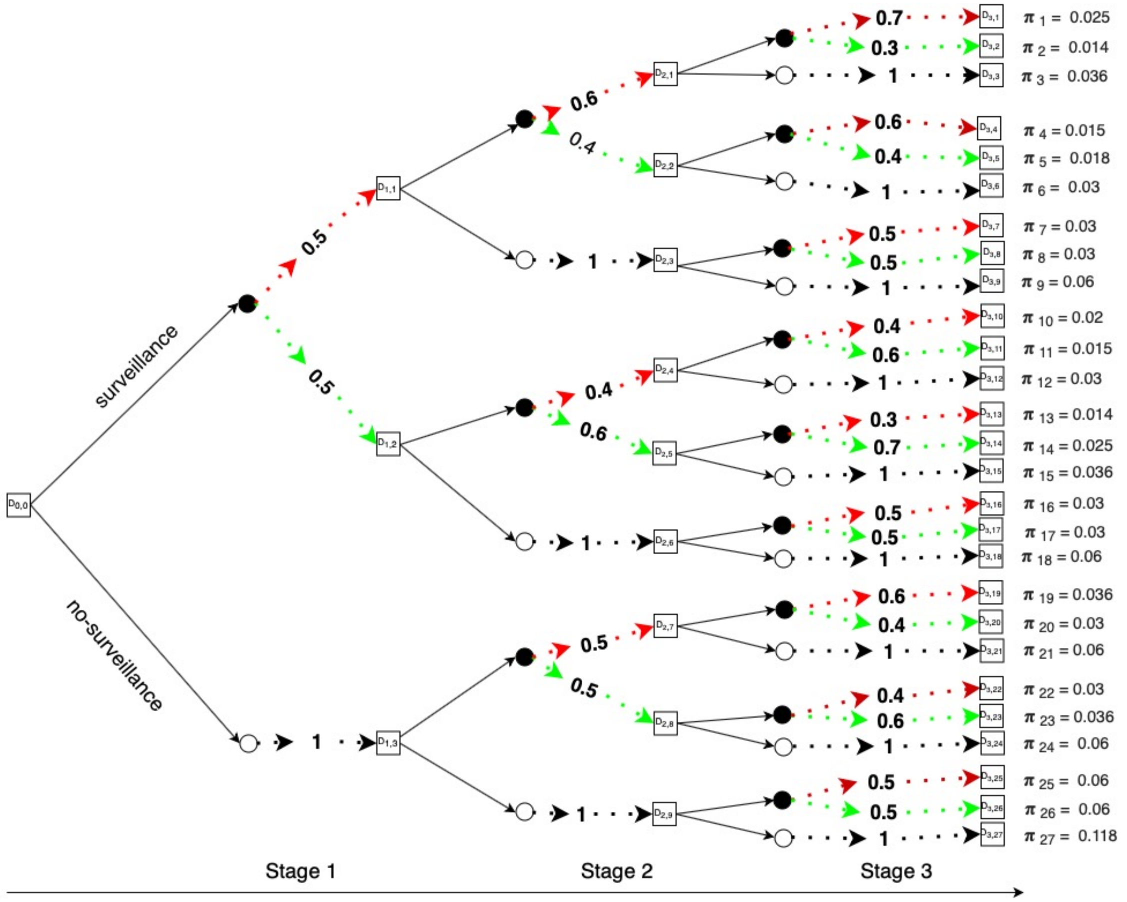


Figure 2.4 Estimating the probabilities of the scenario occurrence for a 3-period example using Algorithm 1. Bold red lines show realizations of high infestation level, dashed green lines show realizations of low infestation level after surveillance, and dotted black lines show realizations of the expected medium infestation level without surveillance. The numbers between lines show the probabilities of scenario realizations. $\pi_1 - \pi_{3t}$ depict the probabilities for each scenario ω.

2.2.4 Notation

Sets and Indices

Γ Set of all sites, $\Gamma = \{1, 2, \ldots, \bar{\Gamma}\}$

K Set of infestation levels, $K = \{1, 2, 3\}$

T Set of time periods, $T = \left\{1, \ldots, \bar{T}\right\}$

Ω Set of scenarios in a scenario tree, $\Omega = \left\{1, \ldots, \bar{\Omega}\right\}$

χ Set of neighboring layers that a spread can happen from a site i; each layer represents a distance dependent neighbor of site i with similar spread rates

i Index for site where $i \in \Gamma$

Θ_i^ι Set of neighboring sites of site i at layer ι

k Index for infestation level where $k \in K$

t Index for time period where $t \in T$

j Index for neighboring sites of site i at layer ι where $j \in \Theta_i^\iota$

ω Index for a scenario where $\omega \in \Omega$

ι Index for neighboring layer where $\iota \in \chi$

Parameters

π_ω Probability for scenario ω

c_1 Cost of surveying (inspecting) a tree

c_2 Cost of treatment

c_3 Cost of removal

α Monetary value of an uninfested tree

ϑ_k Penalty value of each infested tree at infestation level k

r_k Impact rate of each infested tree at infestation level k within a site i, i.e., number of new infestations per infested tree at level k

υ Surveillance efficiency, i.e., percent of infested trees that are identified correctly

τ Discount rate

δ_t Discount factor at time t which is equal to $\dfrac{1}{(1+\tau)^t}$

Ψ_ω Budget for scenario ω

θ_k^ι Infestation impact of k^{th}-level infested trees in neighboring layer ι belonging to site j

κ Maximum number of trees surveyed in each site i

γ_i Number of surveyed trees in site i under surveillance at time t,i.e., $\gamma_i = min\,(N^t_{i\omega}, \kappa)$

$p^{\iota}_{j\to i}$ Probability of infestation spread from site j to i at neighboring layer ι

$\beta^t_{ik\omega}$ Percentage change in belief of infestation after surveillance for site i, infestation level k, at time t, for scenario ω

$\bar{N}_i$ Initial number of tree population at site i

$\bar{I}_{ik}$ Initial number of infested tree population at each infestation level k, at site i

Binary Decision Parameters in Decision Scenario Tree

$$x^t_\omega = \begin{cases} 1 & \text{if surveillance is applied at time } t\text{, for scenario } \omega \\ 0 & \text{otherwise} \end{cases}$$

Decision Variables

$N^t_{i\omega}$ Total number of trees at site i, at time t, for scenario ω

$S^t_{i\omega}$ Number of susceptible trees at site i, at time t, for scenario ω

$\widetilde{I}^t_{ik\omega}$ Believed number of infested trees at site i, at time t, for infestation level k, for scenario ω before surveillance

$\dddot{I}^t_{ik\omega}$ Transition number of infested trees at site i, at time t, at infestation level k, for scenario ω after surveillance without considering total tree population

$I^t_{ik\omega}$ Estimated number of infested trees at site i, at time t, at infestation level k, for scenario ω after surveillance with considering total tree population

$V^t_{ik\omega}$ Number of treated trees at site i, at time t, at infestation level k, for scenario ω

$R^t_{ik\omega}$ Number of removed trees at site i, at time t, at infestation level k, for scenario ω

$H^t_{i\omega}$ Number of trees surveyed at site i, at time t for scenario ω

$Q^t_{ik\omega}$ Number of infested trees remaining after treatment and removal at site i, at time t, at infestation level k, for scenario ω

Linearization Variables

$$u^t_{ik\omega} = \begin{cases} 1 & \text{if transition population is assigned to infestation level } k\text{, at site } i \\ & \text{at time } t \\ 0 & \text{otherwise} \end{cases}$$

2.2.5 Mathematical Model

$$Max \sum_{\omega \in \Omega} \pi_\omega \left(\sum_{t \in T} \delta_t \sum_{i \in \Gamma} \left(\alpha S_{i\omega}^t - \sum_{k=1}^{n} \vartheta_k I_{ik\omega}^t \right) \right) \tag{2.1}$$

Subject to : Initial Total Population

$$N_{i\omega}^1 = \bar{N}_i \qquad \forall\ \omega,\ i, \tag{2.2}$$

Initial Belief of Infestation

$$\widetilde{I}_{ik\omega}^1 = \bar{I}_{ik} \qquad \forall\ \omega,\ i,\ k, \tag{2.3}$$

Transition Infestation Level

$$\ddddot{I}_{ik\omega}^t = \widetilde{I}_{ik\omega}^t \left(1 + x_\omega^t \beta_{iks\omega}^t\right) \qquad \forall\ \omega,\ i,\ t,\ k, \tag{2.4}$$

Susceptible (Healthy) Tree Population

$$S_{i\omega}^t = N_{i\omega}^t - \sum_{k=1}^{n} I_{ik\omega}^t \qquad \forall\ \ \omega,\ i,\ t, \tag{2.5}$$

Population Constraint

$$N_{i\omega}^{t+1} = N_{i\omega}^t - \sum_{k=1}^{n-2} V_{ik\omega}^t - \sum_{k=1}^{n} R_{ik\omega}^t \qquad \forall\ \omega,\ i,\ \&,\ t = 1, \tag{2.6}$$

$$N_{i\omega}^{t+1} = N_{i\omega}^{t} - \sum_{k=1}^{n-2} V_{ik\omega}^{t} - \sum_{k=1}^{n} R_{ik\omega}^{t} + \sum_{k=1}^{n-2} V_{ik\omega}^{t-1} \ \ \forall\ \omega,\ i,\ \&,\ t = 2 \ldots \bar{T} - 1, \tag{2.7}$$

Carrying Capacity Constraints

$$I_{ik\omega}^{t} \leq N_{i\omega}^{t} - \sum_{d=min(k+1,n)}^{n} I_{id\omega}^{t} \qquad \forall\ \omega,\ i,\ t,\ k, \tag{2.8}$$

$$I_{ik\omega}^{t} \leq \ddddot{I}_{ik\omega}^{t} \qquad \forall\ \omega,\ i,\ t,\ k, \tag{2.9}$$

$$\ddddot{I}_{ik\omega}^{t} - I_{ik\omega}^{t} \leq \bar{N}_i \left(1 - u_{ik\omega}^{t}\right) \qquad \forall\ \omega,\ i,\ t,\ k, \tag{2.10}$$

$$\left(N_{i\omega}^{t} - \sum_{d=min(k+1,n)}^{n} I_{id\omega}^{t}\right) - I_{ik\omega}^{t} \leq \bar{N}_i u_{ik\omega}^{t} \qquad \forall\ \omega,\ i,\ t,\ k, \tag{2.11}$$

Number of Treated and Removed Trees

$$V_{ik\omega}^{t} + R_{ik\omega}^{t} \leq I_{ik\omega}^{t} \sum_{a=max[t-k+1,1]}^{t} x_{\omega}^{a} \qquad \forall\ \ \omega,\ i,\ t, \quad k = 1, \tag{2.12}$$

$$R_{ik\omega}^{t} \leq I_{ik\omega}^{t} \sum_{a=max[t-k+1,1]}^{t} x_{\omega}^{a} \qquad \forall\ \ \omega,\ i,\ t \quad k = n-1\ \&\ n, \tag{2.13}$$

Believed (Expected) Number of Infested Trees

$$\widetilde{I}^{t+1}_{i1\omega} = \sum_{g=1}^{n} Q^{t}_{ig\omega} r_g + \sum_{g=1}^{n} \sum_{\iota\in\chi} \sum_{j\in\Theta^{\iota}_{i}} Q^{t\iota}_{jg\omega} \theta^{\iota}_{k} p^{l}_{j\to i} \;\; \forall \;\; \omega,\; i,\; \iota,\; j \;\&\; t = 1 \ldots \bar{T} - 1, \tag{2.14}$$

$$Q^{t\iota}_{ig\omega} = \begin{cases} I^{t\iota}_{ig\omega} - \upsilon V^{t\iota}_{ig\omega} - \upsilon R^{t\iota}_{ig\omega} & g = 1 \\ I^{t\iota}_{ig\omega} - \upsilon R^{t\iota}_{ig\omega} & g = n-1 \;\&\; n \end{cases} \qquad \forall \;\; \omega,\; i,\; t,\; \iota, \tag{2.15}$$

$$\widetilde{I}^{t+1}_{ik\omega} = I^{t}_{i(k-1)\omega} - \upsilon V^{t}_{i(k-1)\omega} - \upsilon R^{t}_{i(k-1)\omega} \; \forall \; \omega,\; i,\; \&\; t = 1 \ldots \bar{T} - 1 \;\&\; k = 2, \tag{2.16}$$

$$\widetilde{I}^{t+1}_{ik\omega} = \left(I^{t}_{i(k-1)\omega} - \upsilon R^{t}_{i(k-1)\omega}\right) + \left(I^{t}_{ik\omega} - \upsilon R^{t}_{ik\omega}\right) \; \forall \; \omega,\; i,\; t = 1 \ldots \bar{T} - 1 \;\&\; k = n, \tag{2.17}$$

Budget Constraint

$$c_1 \sum_{t\in T} \sum_{i\in\Gamma} H^{t}_{i\omega} + c_2 \sum_{t\in T} \sum_{i\in\Gamma} \sum_{k=1}^{n-2} V^{t}_{ik\omega} + c_3 \sum_{t\in T} \sum_{i\in\Gamma} \sum_{k=1}^{n} R^{t}_{ik\omega} \leq \Psi_{\omega} \; \forall \; \omega,\; i,\; t,\; k, \tag{2.18}$$

$$H^{t}_{i\omega} = \gamma_i x^{t}_{\omega} \qquad \forall \; \omega,\; i,\; k,\; and \; t = 1, 2, \ldots, T, \tag{2.19}$$

$$\gamma_i = min\left(N^t_{i\omega}, \kappa\right) \quad \forall\ \omega,\ i,\ t, \tag{2.20}$$

Non-anticipativity Constraints

$$\begin{aligned}
&N^t_{i\omega} = N^t_{i\omega'} \quad S^t_{i\omega} = S^t_{i\omega'} \quad \widetilde{I}^t_{ik\omega} = \widetilde{I}^t_{ik\omega'},\\
&I^t_{ik\omega} = I^t_{ik\omega'} \quad \ddot{I}^t_{ik\omega} = \ddot{I}^t_{ik\omega'} \quad V^t_{ik\omega} = V^t_{ik\omega'},\ \forall\ \omega = \omega' \in \Omega \text{ s.t. } \omega_t = \omega'_t,\ \forall\ i,\ t,\ k,\\
&u^t_{ik\omega} = u^t_{ik\omega'} \quad u^t_{ik\omega} = u^t_{ik\omega'} \quad R^t_{ik\omega} = R^t_{ik\omega'},
\end{aligned} \tag{2.21}$$

Non-negativity and Binary Restrictions

$$N^t_{i\omega},\ S^t_{i\omega},\ I^t_{ik\omega},\ \widetilde{I}^t_{ik\omega},\ \ddot{I}^t_{ik\omega},\ V^t_{ik\omega},\ R^t_{ik\omega}\ \geq\ 0 \quad u^t_{ik\omega} \in \{0,1\}\ \forall\ \omega,\ i,\ k. \tag{2.22}$$

The objective function of our model, as can be seen in Equation (2.1), maximizes the expected number of susceptible (healthy) trees in the managed area over the planning horizon minus the penalty associated with the number of asymptomatic and symptomatic infested trees present in the area over a set of plausible infestation scenarios. Equations (2.2) and (2.3) initialize the size of the host tree population and the expected infestation levels for each site i and scenario ω in the model at time period 1. Equations (2.6) and (2.7) estimate the total size of the ash tree population and its change over time as more infested trees are removed and treated. Based on the expected infestation level and decisions taken at the surveyed sites, we estimate the number of removed or treated trees, which are infested, at a given site i from the total tree population at that site. The treated trees that are removed from the infested tree population will be moved back to the infested population at later periods once they become susceptible after a two-year immunity period.

Equation (2.4) computes the realizations of infestation scenarios at each level k after the surveillance decisions. We assume that after surveillance, the estimated number of infested trees for each infestation level k is known. Term $\widetilde{I}^t_{ik\omega}$ represents the believed expected number of trees infested at a particular level k. If surveillance is applied, the parameter x^t_ω will take a value of 1, and the value of $\widetilde{I}^t_{ik\omega}$ will change by $\beta^t_{ik\omega}$, which defines the percent change in the expected level of infestation after the surveillance is performed. Thus Equation (2.4) estimates the number of infested trees for each infestation level k after surveillance where $\dddot{I}^t_{ik\omega}$ denotes the transition number of the infested trees at each infestation level k, which represents the estimated number without considering the total tree population in a site i.

Equation (2.5) calculates the number of susceptible (i.e., uninfested and treated) trees. The total population in constraint (2.5) with Equation (2.7) also ensure that treated trees are moved back to the infested tree population after their immunization period ends.

We also need an equation to calculate the estimated number of infested trees in a tree population because the number of new infestations is limited by the maximum susceptible tree population that could be infested, i.e.:

$$I^t_{k\omega} = min \left(N^t_{i\omega} - \sum_{d=min(k+1,n)}^{n} I^t_{id\omega}, \dddot{I}^t_{ik\omega} \right) \tag{2.23}$$

Equation (3.45) defines the estimated number of infested trees at a survey site i, for each scenario ω, time period t, at infestation level k and implied that the estimated number of infested trees at an infestation level k cannot exceed the total number of healthy trees minus the infested trees at higher infestation levels $(k+1, \ldots, n)$ at a site

i. If the remaining size of the healthy tree population exceeds the number of infested trees calculated with Equation (2.4), the estimated number of trees is set equivalent to the transition number of infested trees that gives the estimated number of trees based on infestation growth equations and realization of the uncertain infestation outcomes after surveillance. Equations (2.8)–(2.11) provide an equivalent linearization of the non-linear carrying capacity constraint (3.45). Equations (2.8) and (2.9) set an upper bound on the estimated number of infested trees and Equations (2.10) and (2.11) set a lower bound by using an auxiliary binary decision variable $u^t_{ik\omega}$.

Equations (2.12) and (2.13) define the upper bound on the number of treated and removed trees. As the infestation spreads, more susceptible trees could become infested. When the budget is too small to treat or remove all detected infested trees immediately, the infestation spreads to other trees in a given site and to neighboring sites $j \in \Theta^\iota_i$.

Equation (2.14) defines the expected number of newly infested trees at level $k = 1$ in time period $t + 1$. The term, $Q^t_{ig\omega}$, denotes the number of untreated or unremoved trees in time period t in site i and is defined in Equation (2.15). Parameter υ defines the surveillance efficiency in Equation (2.15) which is the proportion of infested trees that are detected at a site after surveillance. Term $p^\iota_{j\to i}$ in Equation (2.14) defines the probability of infestation spreading from site j to site i located at the ι^{th} distance class from j. To calculate the infestation spread to a site i we used four distance-dependent spread layers, ι, which cover a 4-km radius from the infested site with 1-km distance intervals. Term Θ^ι_i defines the spread rate from the neighboring sites at each distance class ι to a given site i.

Inspecting a small sample of trees allows detecting only a portion of the infested trees and leads to the failure to treat or remove all the infested trees at a survey site. This issue is handled by Equations (2.16) and (2.17). Constraint (2.16) ensures that trees at higher infestation levels ($k > 1$), if not removed or treated, transition to the next infestation level, $k+1$ at the next time period. Coefficient υ in Equation (2.16) adjusts the number of removed and treated trees to compensate for the uncertainty of the surveillance outcomes.

Equation (2.17) states that trees that reach the highest infestation level $k = n$ are considered dead and remain at this level until the end of the planning horizon. We assume that dead trees do not pose an infestation threat but should be removed due to hazard and liability concerns if the budget allows.

The medium extent (M) represents the expected level of infestation in the absence of surveillance based on prior information about the invasion and is calculated in the model using Equations (2.14) - (2.17). Because the surveillance selection binary parameter x_ω^t in Equation (2.4) becomes 0 under no surveillance, the expected infestation level will not change. When surveillance occurs, the $x_\omega^t = 1$, and so the expected infestation level will change by $\beta_{ik\omega}^t$, (i.e., by +0.4 or -0.2 in realizations of (H) or low (L) infestation levels, respectively).

Equation (2.18) sets an upper bound on the available budget over the planning horizon in a scenario ω. Treatment and removal decisions depend on the infestation level k. Term $H_{i\omega}^t$ denotes the number of inspected trees and is defined in Equation (2.19). We assume that only a sample of maximum κ trees is inspected at each surveyed site. In Equation (2.20), γ_i defines the number of surveyed trees, which is the minimum of the total number of trees in site i and sample size κ. Terms $V_{ik\omega}^t$ and $R_{ik\omega}^t$ denote the number of treated and removed trees, respectively.

Equation (2.21) lists the non-anticipativity constraints, which ensure that the scenarios with the same history up to a given stage t share the same decisions until that stage. For example, if two scenarios $\omega^{'}$ and ω have the same history of infestation and surveillance decisions until period t, i.e., $\omega_t = \omega_t^{'}$, all decision variables up to stage t should be equal to each other. Finally, Equation (2.22) defines the non-negativity constraints on the decision variables and a binary status of the linearization variable, $u^t_{ik\omega}$.

2.3 Model Application and Data

We applied our MSS-MIP model to assess the surveillance and management options for EAB infestation in Winnipeg, Manitoba, Canada. The study area was divided into 1x1-km survey units. We estimated the ash density at each survey site from a municipal tree inventory (City of Winnipeg. (2018), H. Daudet, City of Winnipeg, Urban For. Br., pers. comm.), which provided information about tree species, ownership, and size (Figure 2.5).

We estimated the probabilities of EAB spread from historical records of urban EAB infestation in Twin Cities, Minnesota [Osthus, 2017]. As with Winnipeg, we divided the Twin Cities area into a grid of 1x1-km potential survey sites. For each $1 - km^2$ site j, we estimated the distance to the nearest infested site in a particular year and, based on that distance, estimated the infestation likelihood for a corresponding distance 1-km class. Using a set of distance-dependent infestation probabilities and the locations of the infested sites, we then generated the likelihoods of EAB spread for potential survey sites in Winnipeg. When calculating the likelihood of new infestations at a site, we also have taken into account the chance of the infestation spread from the surrounding sites. For each survey site, the model tracked the potential spread of EAB from the neighboring sites at four 1-km distance classes each 1, 2, 3, and 4 km away from the infested site. Distance-dependent probabilities of spread originating from symptomatic infested ash trees were estimated as 0.34 at the infested site, 0.21, 0.12, and 0.05 at the neighboring sites at 2, 3 and 4-km distance while those originating from infested asymptomatic ash trees in 1, 2, 3 and 4-km neighboring sites are 0.2, 0.15, 0.08 and 0.03, respectively. We estimated two sets of distance-dependent probabilities of spread for asymptomatic

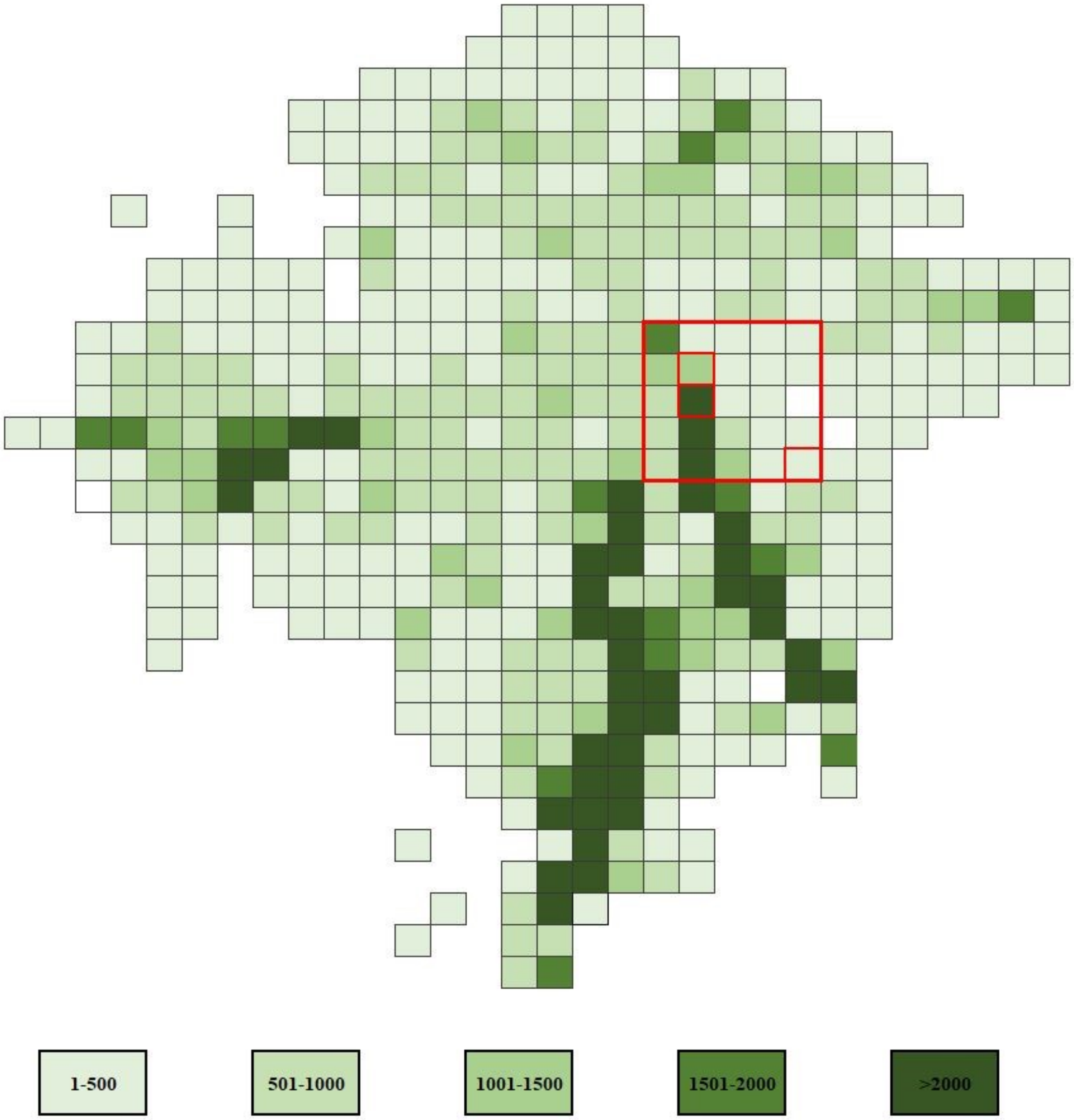


Figure 2.5 Host tree density map for a case study area in Winnipeg, Canada. Square outline delineates the neighborhood area around the current EAB infestations.

and symptomatic infested trees. Symptomatic trees have typically more EAB galleries than asymptomatic trees, hence their threat to other susceptible trees was assumed to be higher [Knight et al., 2012]. We assumed that dead trees do not produce propagules and have no impact on other susceptible trees. To address the uncertainty about EAB spread, we modeled changes in the anticipated levels of infestation once the surveillance is completed. We assumed that the manager anticipates changes in the infestation levels after surveillance by -20% or +40% from its current level. Each of these two realizations occurs with the probability of 0.5 in the first period and then updated dynamically for all other periods using Algorithm 1 as shown in Section A.2. When no surveillance is applied, trees progress to the next infestation level based on the default assumption about the infestation level.

The efficiency of surveillance depends on the choice of the method to detect the signs of EAB attack. In Canada, two common inspection methods used in precious survey campaigns include sampling host tree branches and installing sticky traps [Hopkin et al., 2004, Ryall et al., 2011, 2013, Turgeon et al., 2016]. Sampling branches and peeling their bark to inspect for EAB galleries is the most reliable method to detect EAB [Ryall et al., 2011, Turgeon et al., 2016]. The application of sticky traps includes hanging the traps baited with plant volatile and EAB pheromone followed by one-two checkup visits [Ryall, 2015]. Based on the previous EAB survey study [Yemshanov et al., 2019b], the detection rate for branch sampling was set to 0.7, based on a typical sample of two mid-crown branches from a medium-sized tree [Ryall, 2015]. The likelihood of a single sticky trap detecting the presence of an EAB population on a tree was set to 0.5. The specified detection rates were determined for urban EAB populations in southern Ontario, Canada but should be applicable for Winnipeg given its tree size distribution is typical of other urban areas in Canada. Based on the experiences from past EAB surveys in Ontario typical sampling rates for branch sampling and trapping rarely exceeded 5-10 trees $-km^{-1}$ due to the high cost of inspections, hence we consider the scenarios using a fixed sampling rate $\kappa =$ 5 trees $-site^{-1}$. We used the survey cost estimates from Yemshanov et al. [2019b]: $87 for installing a sticky trap and $124 to inspect a tree via branch sampling. The monetary value of services provided by a host tree was estimated at 72 CAD [Kıbış et al., 2021]. The cost of treating asymptomatic infested trees was estimated at 180 CAD and the cost of removing an infested tree was estimated 800 CAD [Yemshanov et al., 2019b]. We explored the scenarios with the budget limits ranging from 1M to 2M over a 5-year planning horizon. The social discount rate was set to 2%.

2.4 Results

Our results reported below describe the general model behavior and present outputs which have practical utility for decision-making, including general indication where and when treatments and/or tree removal may be feasible, differences between using trapping and branch sampling methods, and the impacts of survey timing and management actions.

2.4.1 Optimal Management

Cost of Surveillance, Treatment, and Removal We illustrate the general model behavior by showing the solutions for a small area surrounding the infested sites - a 5x5 subset of survey sites under a budget of $2M (red outline in Figure 2.5). Since each scenario of the scenario tree has a different combination of the infestation probabilities, management decisions, and budget allocations for surveillance, treatment, and tree removal, we present examples of optimal solutions for four distinct scenarios. For each scenario we assign an identifier that lists the occurrence of surveillance events and the extent of the detected infestation over a 5-year planning horizon. For example, after surveillance is done, a possible outcome of the survey is the detection of high (H) or low (L) levels of infestation. In the absence of surveys, the expected state of the infestation is a medium extent (M). Our scenarios depict different levels of infestation and distinct timing of surveillance. For example, scenario H-H-H-H-H implies that surveillance is done each year and a high infestation level is detected. Meanwhile, scenario 126 with realization L-L-M-H-H implies that in the first two years surveillance is done and a low infestation level is detected. Note that H and L only occur under surveillance, and M only occurs under no surveillance. Thus, we do not need to provide a specific notation for surveillance decisions to represent scenarios. In the third year, no surveillance is done, so M means we still remain truthful to the expected infestation growth calculated by the mathematical model based on the initial belief of infestation of the manager. And lastly, the last two years, surveillance is done, and a high realization is encountered. In our case, the scenario with the highest net benefits is scenario 161, which only surveys in the first stage of the problem and has a low infestation realization in the initial stage, as denoted by L-M-M-M-M.

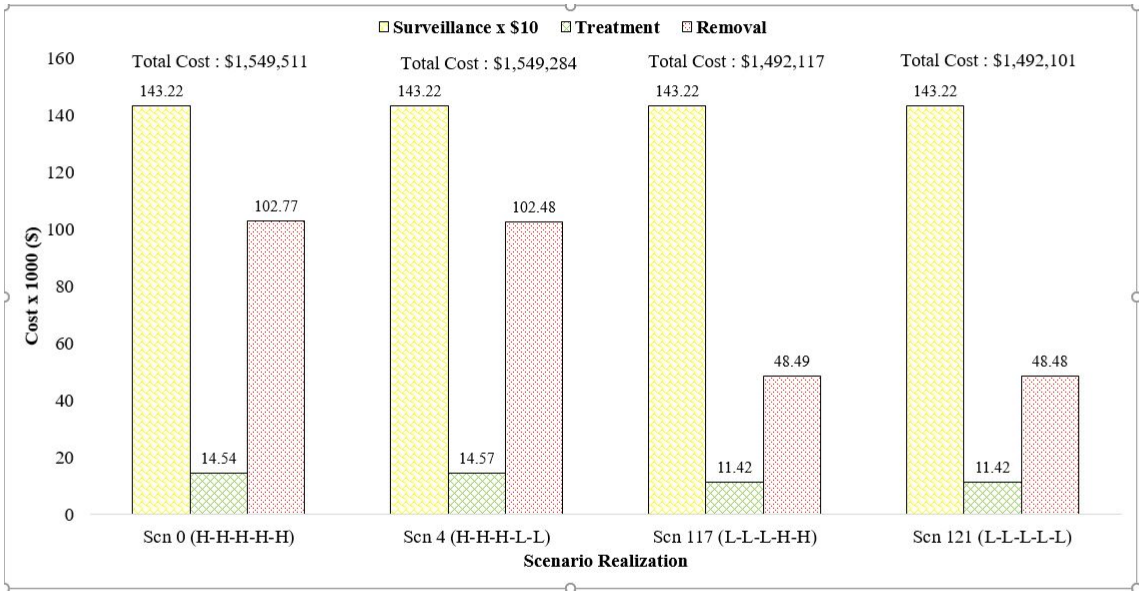


Figure 2.6 Treatment, removal, and surveillance cost for low and high realizations over 5 years for four different scenarios. Scenario 0 is H-H-H-H-H, scenario 4 is H-H-H-L-L, scenario 117 is L-L-L-H-H and lastly, scenario 121 is L-L-L-L-L, where L and H stand for low and high realization, respectively.

In Figure 2.6, scenarios H-H-H-H-H and L-L-L-L-L assume the detection of correspondingly high and low infestation levels in all time periods. Scenario H-H-H-L-L assumes the detection of high infestation levels in periods 1-3 and low infestation levels in periods 4 and 5 and scenario L-L-L-H-H depicts an opposite survey outcome when low infestation levels are detected in periods 1-3 and high levels in periods 4 and 5. The total cost in each scenario varies due to different streams of treatment and removal decisions. Given that the surveys occur every time period, the cost of surveillance is the same for all scenarios however, the cost of treatment and tree removal depends on the level of the detected infestation. For example, scenario L-L-L-L-L had the lowest total cost because the infestation was detected at a low level and required less treatment and removal efforts. The level of infestation detected early has a higher impact on management actions and their total cost. More budget was allocated in the scenario with the detected high level of infestation in first two periods than in the scenario with the detected low infestation level in periods 1-2 (compare the costs of scenario H-H-H-L-L and scenario L-L-L-H-H in Figure 2.6).

Where to Survey, Treat, and Remove Most of the applied survey and management actions in optimal solutions have occurred in close proximity to the infested sites hence we illustrate the model behavior using a 5x5 site area proximate to the infested sites (big red rectangle in Figure 2.5). Figure 2.7 shows the number of infested, treated, and removed trees for time periods 1-4 in the worst-case scenario H-H-H-H-H with the high level of infestation detected in all periods. When no action is taken, more trees get infested in close proximity to the infested sites, which increases the local rate of spread and the number of infested trees. Timely treatment and tree removal actions help reduce the number of infested trees in close proximity to the infested sites. Most of the treatment occurred in the sites with the detected infested trees in the first two periods (when treatment is the most effective). Tree removal was prescribed roughly in the same selected locations but occurred over periods 1-4. All treatments and tree removals in period 1 occurred in the sites with the original infestations. Imperfect detection in the first period required a more aggressive treatment and removal in the following periods to compensate for poor detection accuracy. Our results show that detecting and treating the infested trees as early as possible is the most cost-effective approach, but imperfect detection necessitates a larger-scale treatment and tree removal campaign in the following periods.

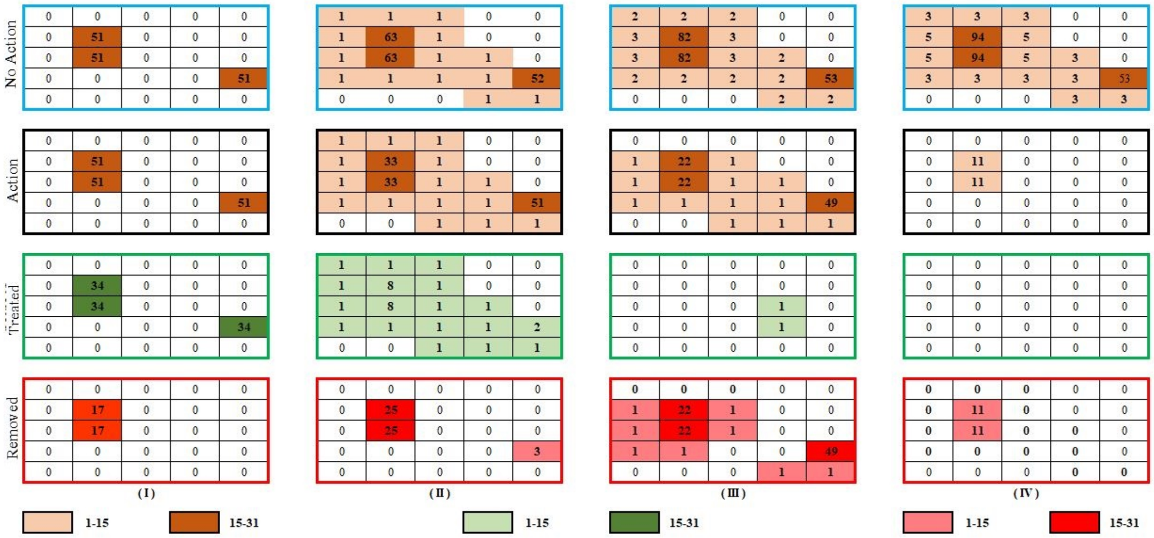


Figure 2.7 Total number of infested trees under no action, infested trees under optimal action, treated, and removed trees for each period for scenario H-H-H-H-H (high infestation level detected in all periods) over planning periods 1-4.

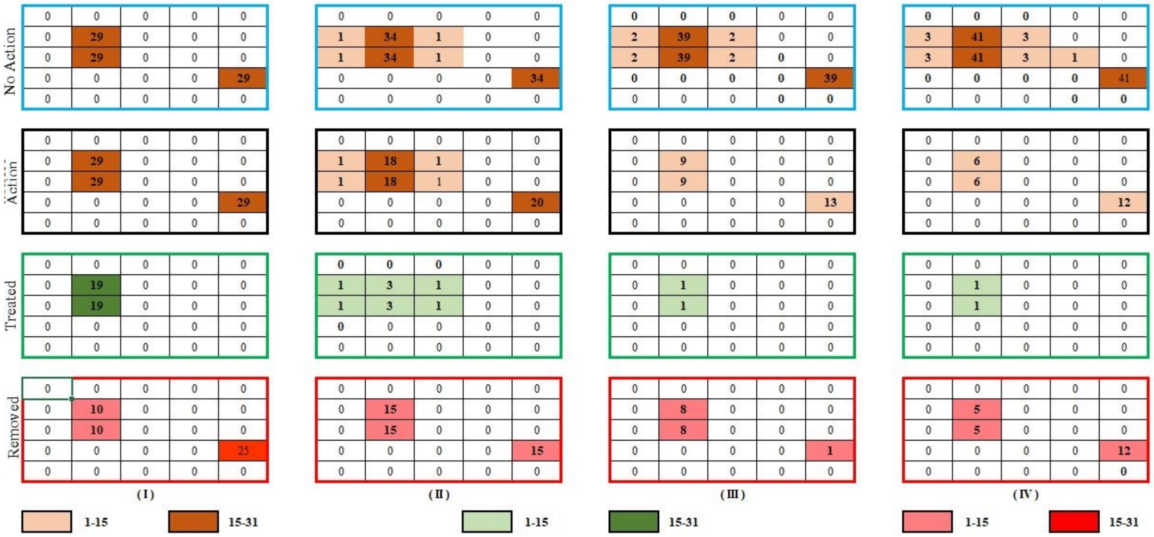


Figure 2.8 Total number of infested trees under no action, infested trees under optimal action, treated, and removed trees for each period for scenario L-L-L-L-L (low infestation level detected in all periods) over planning periods 1-4.

Figure 2.8 reports similar results but for scenario L-L-L-L-L when the surveillance detects the low infestation level in each time period. Compared to the worst-case scenario, the detection of the infested trees occurred at shorter distances from the infested sites. Treatment and removal mostly occurred in the currently known infested sites.

We have also compared the impacts of taking no action in four different scenarios, H-H-H-H-H, H-H-L-L-L, L-L-H-H-H, and L-L-L-L-L. We define the net benefit summary metric as a difference between the objective function value (which is the total value of healthy and treated trees in a landscape) and the cost of surveillance, treatment, and removal. As Table 2.1 suggests, optimal management solutions with treatment and tree removal have higher benefits than no-action solutions, which indicates that active treatment and removal options remain cost-effective despite the incurred survey and management costs.

Table 2.1 Net Benefits in the Optimal Management (Action) and No-Action Solutions for Scenario H-H-H-H-H, H-H-L-L-L, L-L-H-H-H, and L-L-L-L-L

Scenario	**Action**	**No Action**
H-H-H-H-H	\$122,258,000	\$121,704,000
H-H-L-L-L	\$122,261,590	\$122,183,550
L-L-H-H-H	\$122,399,610	\$122,327,760
L-L-L-L-L	\$122,400,000	\$122,119,000

2.4.2 Effect of Surveillance Timing on Net Benefits

Surveillance is crucial in our model as no action can be taken without surveillance. When no surveillance is performed in a particular time period the infestation continues to spread and cases larger damage. We illustrate the importance of maintaining the surveillance regime by analyzing the optimal solutions with a distinct timing of survey actions. We compare four scenarios, H-M-M-M-M, M-H-M-M-M, M-M-H-M-M, and M-M-M-H-M, which apply the surveys only in time periods 1, 2, 3 or 4. All scenarios have the same surveillance costs (i.e., one year of surveys only). Table 2.2 indicates that the scenarios with the earliest survey actions have the lowest total treatment and removal cost. This is because early detection leads to a more effective treatment or removal when the infestation is at an early stage. When the surveillance is delayed, infestation is allowed to spread to further distances, and more trees get infested and will require treatment or removal. In particular, the efficacy of tree removal action is affected by the timing of surveillance. The sooner the infestation is detected the less it will cost to remove the infested trees.

Table 2.2 Costs of Surveillance, Treatment, and Removal for the Scenarios with Different Timing of the Survey Actions

Scenario	Survey Period	Surveillance cost	Treatment cost	Removal cost	Total cost	Net Benefits*
H-M-M-M-M	1 (no-delay)	$286,440	$12,100	$94,520	$393,059	$123,990,000
M-H-M-M-M	2	$286,440	$5,640	$134,000	$462,082	$123,130,000
M-M-H-M-M	3	$286,440	$6,340	$160,680	$453,457	$123,170,000
M-M-M-H-M	4	$286,440	$3,620	$175,660	$465,724	$123,030,000

* The net benefit value is calculated as a difference between the objective value minus the total cost of surveys, treatment and tree removal.

The scenario with no survey-delays (H-M-M-M-M) also had the highest net benefits. Overall, the delay in surveillance allows the infestation to spread to a larger area and will necessitate costlier tree removal and treatment actions, hence it is always beneficial to survey and treat the sites as early as possible.

2.4.3 Budget Allocation and Priority of Actions

We have solved the problem for the range of budget levels including $1.45M, $1.5M, $1.75M, and $2M. Here, we present optimal solutions for treatment and removal decisions in the worst-case scenario H-H-H-H-H, with the detected high levels of infestation and the surveillance done in every time period. Figure 2.9 shows the surveillance, treatment, and removal costs for the worst-case scenario at different budget levels. At the lowest budget, $1.45M, only a portion of the infested trees in closest proximity to the infested sites can be treated and also a small number of symptomatic trees can be removed in period 1. Since the budget is too small to treat or remove all detected infested trees, the infestation continues to spread. At the budget level of $1.75M, more funds are available for treatment of asymptomatic trees and removal of symptomatic and dead trees. Given a small extent of the current EAB infestation in Winnipeg, all the detected infested trees in close proximity to the infested trees can be treated, but only a small portion is feasible to remove due to the high cost of tree removal. The best strategy is to treat the detected infested asymptomatic trees, and if funds permit, remove a portion of symptomatic infested and dead trees in closest proximity to the infested sites.

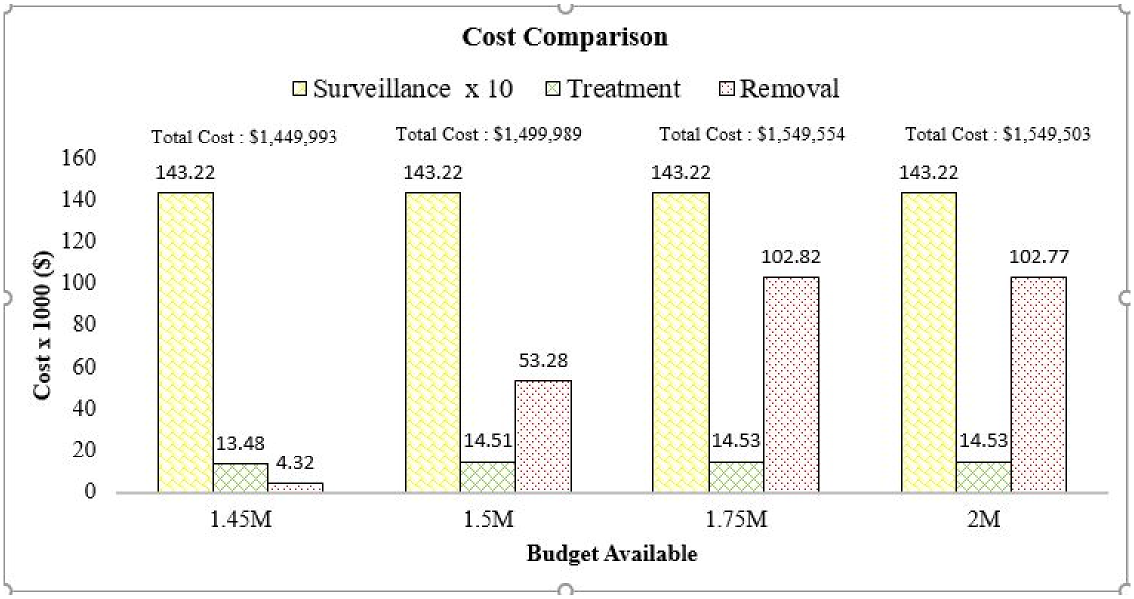


Figure 2.9 Surveillance, treatment, and removal costs for different budget levels under worst-case scenario H-H-H-H-H with continuous surveillance over the planning horizon and the detected high level of infestation.

The budget level affects the timing and the extent of tree removal and treatment actions. Figures 2.10 (a) and (b) show the total number of infested, treated, and removed trees over a planning horizon for budget allocations of \$1.5M and \$2M, respectively. In both solutions, treatments of asymptomatic infested trees and removal of symptomatic infested trees help minimize the impact of infestation on a host tree population. Treatments of healthy and asymptomatic trees prevent them from transitioning to a more severe infestation level and so helps reduce the local spread rate in the next time period. Removal of symptomatic infested trees has a higher priority than removal of dead trees because it reduces the chances of EAB spreading to nearby trees in close proximity to the infested nuclei.

For example, in the scenario with \$1.5M budget, insufficient budget level limited the scope of tree removal actions to a few symptomatic infested trees (Figure 2.10). Note that the removal of dead trees may be mandatory in an urban setting in some circumstances due to liability or safety concerns, and therefore may require additional funds to be allocated for mandatory tree removal. Given a small extent of the current EAB infestation in Winnipeg, we did not include mandatory tree removal options but this could be a focus of future work if the EAB infestation causes widespread tree mortality (like in previous urban EAB outbreaks in Ontario and Michigan). Treatments of asymptomatic trees is only effective in the first three time periods when the infestation is in its early stage. Removal of symptomatic trees becomes most effective in period 2 and essentially is a more preferred action than the treatment as the number of symptomatic trees increase. Removing dead trees is feasible in periods 2 and 3 only when there the budget is sufficient to treat the detected asymptomatic and remove the symptomatic trees. No management action is taken for periods 4 and 5 under a budget of \$1.5M because all budget was already spent during time periods 1, 2, and 3.

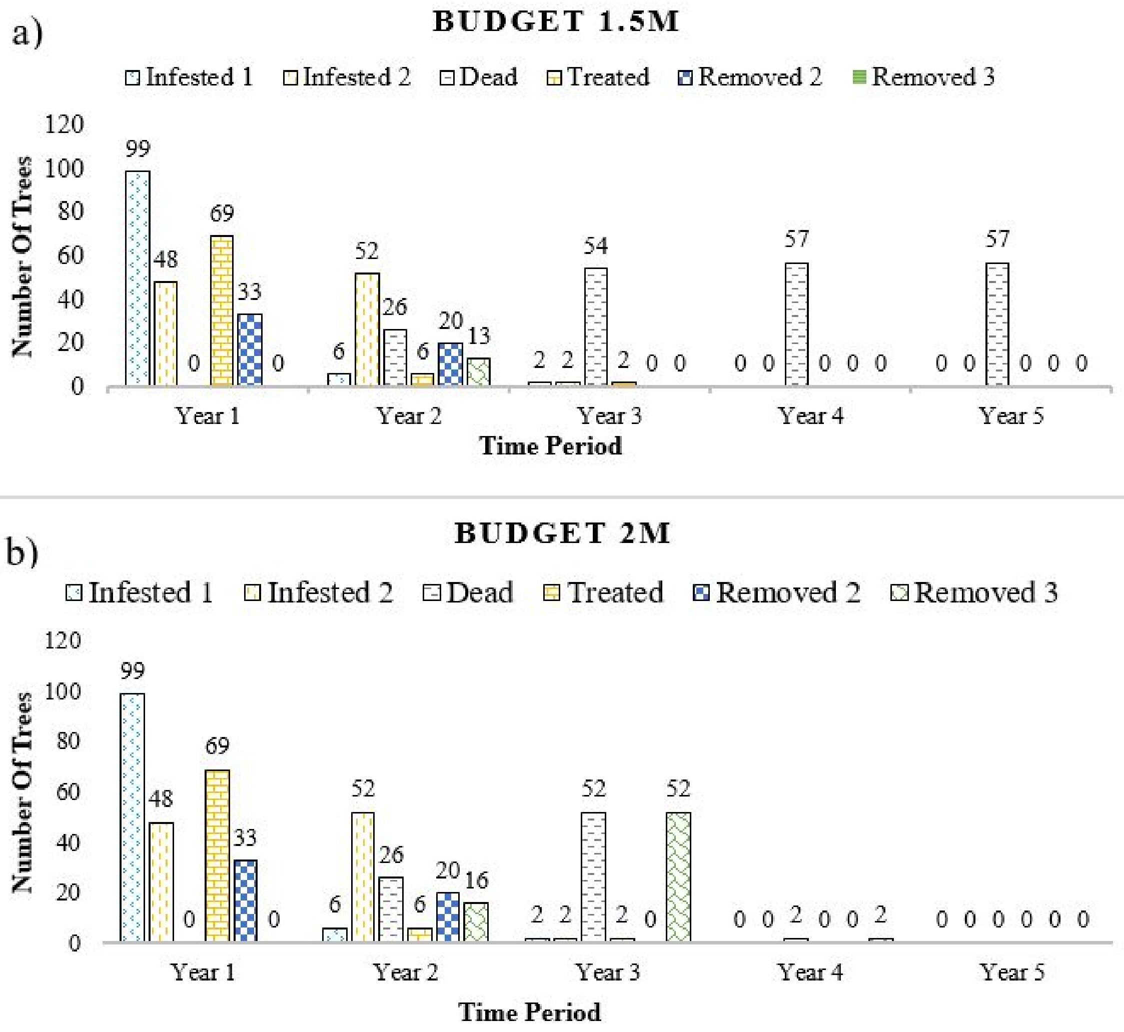


Figure 2.10 Total number of infested, treated, and removed trees over time under scenario 0 with \$1.5M (a) and \$2M (b) budget.

2.4.4 Is treatment a choice?

We also illustrate the impact of applying the treatments of asymptomatic trees using four distinct scenarios with different infestation level sequences of management decisions. Table 2.3 shows differences in net benefit values between the solutions with and without treatments in the scenarios having different levels of infestation. The scenarios show distinct infestation profiles with the surveillance done in all time periods. In general, treatments increased the net benefit value in all infestation scenarios.

Table 2.3 Net Benefit Values in Optimal Treatment vs No-Treatment Under Various Scenarios

Scenario	Scenario Description	No-Treatment Net Benefits	Optimal Treatment Net Benefits
H-H-H-H-H	1 (no-delay)	\$121,994,000	\$122,258,000
H-H-L-L-L	2	\$122,184,000	\$122,262,000
L-L-H-H-H	3	\$122,328,000	\$122,400,000
L-L-L-L-L	4	\$122,335,000	\$122,400,000

2.4.5 Impact of Surveillance Efficiency

Surveillance efficiency has a direct impact on the total cost, therefore, affecting also the total net benefits. Here we discuss the trade-off of the surveillance cost and surveillance efficiency.

We have compared the optimal solutions using branch sampling vs. the detection with sticky traps in a high-infestation scenario H-H-H-H-H with the surveillance done in all time periods 1-5. Figure 2.11 shows that branch sampling yields a higher objective value than using sticky traps because it has a higher detection accuracy and so enables finding and treating more infested asymptomatic trees at early stages. However, branch sampling is a more costly option hence the total cost portion spent on surveillance is higher (Figure 2.12). Comparatively, the solutions with sticky traps spend more on treatment and a significantly more on removal because sticky traps cannot detect the infestation at early stages. We have also compared the optimal timing of treatment and tree removal actions for the solutions using branch sampling and sticky traps in high-infestation scenario H-H-H-H-H under \$2M budget limit. Figure 2.13 shows the number of infested, treated, and removed trees over time for each of the detection methods. Both methods show a similar number of removed and treated trees in period 1, but the solutions with branch sampling show that a lower number of trees required treatment and removal in periods 2 and 3. This is because the higher accuracy of the branch sampling method allows

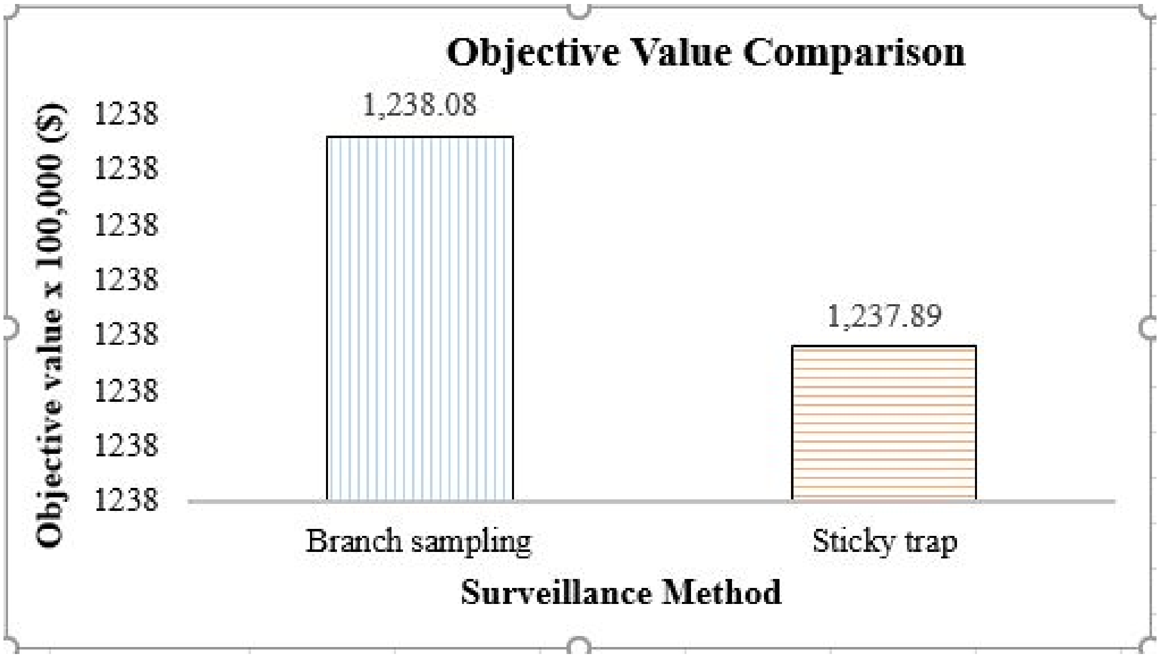


Figure 2.11 Objective function values for scenario 0 (H-H-H-H-H) using branch sampling and sticky trap methods with $2M budget.

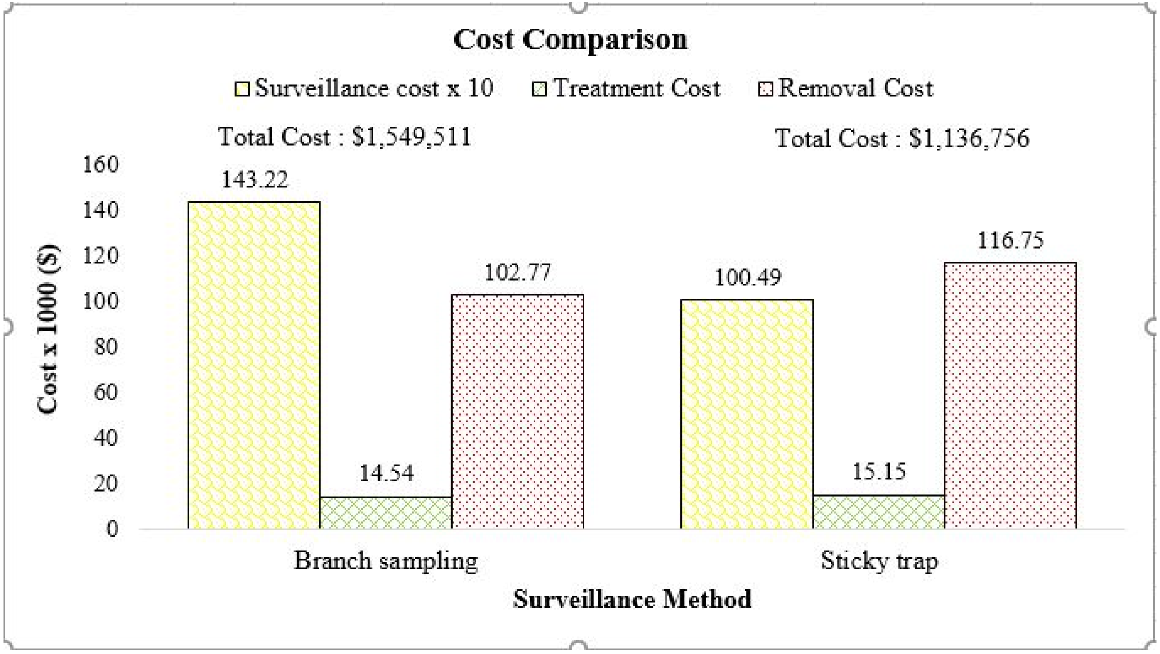


Figure 2.12 Surveillance, treatment, and removal cost comparison between branch sampling and sticky trap methods for scenario 0 (H-H-H-H-H) with $2M budget

detecting more infested trees at the earliest possible time (period 1) and so fewer infested trees will need treatment or removal in the following periods. Also, the solutions using sticky traps required more periods of surveys to detect the bulk of the infestation than the solutions using branch sampling. This is because sticky traps have lower surveillance efficiency. Detecting and subsequently treating or removing fewer trees in a current period leads to a higher infestation rate in the following periods. Our results indicate that while branch sampling has a higher cost, its higher efficiency makes up for the increased cost in terms of reducing the number of trees infested with EAB.

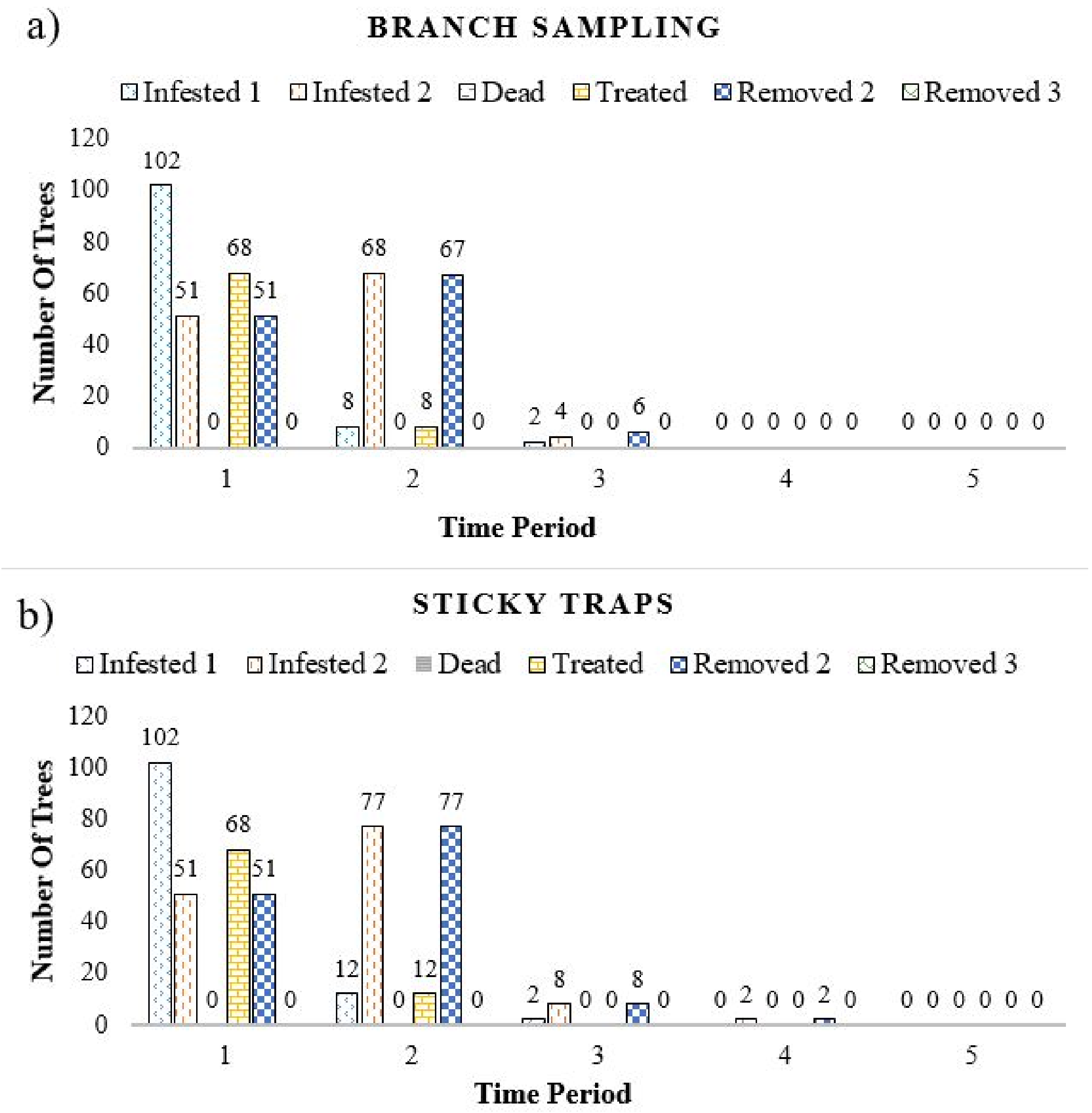


Figure 2.13 Total number of infested, treated and removed trees over time for scenario 0 with budget $2M: a) using branch sampling; b) using sticky traps.

In terms of the net benefits, the surveillance via sticky traps costs less than using branch sampling. However, branch sampling has a higher objective value, that is, it enables keeping more uninfested trees in the managed area. The difference in net benefit values is bigger in small-budget solutions than in large-budget solutions (Table 2.4) because small budgets are insufficient to treat and remove all the detected trees which essentially renders the survey efforts ineffective.

Table 2.4 Net Benefit Values in the Solutions Using Branch Sampling and Sticky Traps

Surveillance Method	**$1.45M**	**$1.5M**	**$1.75M**	**$2M**
Branch Sampling	$121,913,000	$122,055,000	$122,257,000	$122,258,000
Sticky Traps	$122,652,000	$122,652,000	$122,481,000	$122,652,000

2.5 Discussion and Conclusions

In this chapter, we modified the MSS-MIP model proposed by Kıbış et al. [2021] to evaluate main management strategies to manage an EAB infestation in Winnipeg, Canada. The model applies surveillance to inform decisions on optional treatment and removal of the infested trees under a limited budget. The sequences of scenario decisions and associated infestation outcomes are integrated into a scenario tree where, for each scenario, the probability of infestation in a given time period is calculated dynamically. Managing pest outbreaks often requires allocating scarce resources between surveillance and control actions. Our approach helps address this challenge and demonstrates how accounting for key assumptions about the infestation severity, cost, and spatial patterns of the infestation may affect the scope and timing of control decisions. Our results demonstrate that timely detection and early response actions are key factors in the successful control of a pest invasion and a maximization of net benefits of the urban ash trees.

The findings also emphasize the importance of treatments of ash trees as early as possible. Treating asymptomatic trees at the earliest stages of invasion provides higher net benefits than tree removal or no-treatment options. However, we show that treatment of the infested trees is only effective when done at the earliest stage of infestation. When the surveillance is delayed tree removal becomes a more preferred option but would require significantly higher cost to achieve the same level of control.

Our results also provide new insights about the preferred use of trapping versus branch sampling techniques for EAB detection. In multi-year EAB surveys, the use of branch sampling is advised because it yields better accuracy of detecting the infested asymptomatic trees and so, when implemented at early stages of infestation, enables treating and removing more infested trees which may help reduce the local rate of spread.

Another important insight from our work is that the level of available budget essentially controls the decisions on treatment or removal of the infested trees. Since tree removal is costly the extent of tree removal actions depends on the level of the budget available after the surveillance. Our results emphasize the importance of allocating a sufficient budget for tree removal to slow the spread of EAB. Tree removal becomes less important in small-budget solutions where the optimal policy is to spend most of the budget on treatments.

Future work

Dealing with uncertainty about the pest's rate of spread is a common problem in managing biological invasions. Precise estimates of spread rates are rarely available for newly detected infestations and can only be approximated from previous infestations or from general knowledge of the invasive organism's biology. In our model, we used a simplified treatment of the uncertainty about the infestation rates via an introduction of the expected high and low levels of infestation. Potentially more infestation levels could be added to a scenario tree in a future study. Refining the data assumptions is another avenue for improving the practical utility of the model. For instance, colder winters in Winnipeg may cause EAB to switch to a longer, 2-year life cycle, which would require the re-parameterizing of the model for two-year time steps.

In our model, we assumed a fixed tree sampling rate across all surveyed sites. Potentially, the problem could be modified by allowing an optimal selection from a set of pre-defined sampling rates via an introduction of auxiliary binary variables which select a particular sampling rate value at a survey site (and a linked set of surveillance efficiencies). Further refinements also include an account for long-distance spread assumptions for distances beyond 4 km. These modifications are expected to significantly increase the problem size and its combinatorial complexity and will require new approaches to solve the model for practical cases. Another future direction includes the introduction of coherent risk measures to capture the variability of the random variables in the upcoming stages of the model enabling the risk-averse decision making of the manager. This will be the focus of our future work.

CHAPTER 3

RISK-AVERSE MULTI-STAGE STOCHASTIC OPTIMIZATION FOR SURVEILLANCE AND OPERATIONS PLANNING OF A FOREST INSECT INFESTATION

3.1 Introduction

Multi-stage stochastic programming has been widely used in many fields, including but not limited to healthcare [Yin and Büyüktahtakın, 2020], forestry [Kıbış et al., 2021], and finance [Abdelaziz et al., 2007]. Multi-stage stochastic programs typically minimize (maximize) an expectation criterion that calculates the expected cost (benefit) of all possible scenarios, each of which is mapped with a certain probability of occurrence. The expectation objective is useful in situations where the uncertainty does not indicate a potential for observing extreme events. However, if the environment features the possibility of experiencing high-impact events, even with small probabilities, the expectation criterion may not perform well because it does not capture the variability of events. In these situations, where a high impact scenario might happen, the expectation criterion is accompanied by a risk measure.

In this research, we consider a risk-averse multi-stage stochastic mixed-integer program (RA-MSS-MIP), where the objective function is a combination of an expectation operator and a Conditional Value-at-Risk (CVaR) measure for each stage. We then apply this model to optimize the surveillance and control of a non-native forest insect, the emerald ash borer (EAB), which has infested large areas covered with ash trees in North America. The EAB is a wood-boring beetle native to Asia and discovered in the United States in 2002. Since its discovery, the EAB has killed millions of ash trees and cost homeowners and local governments billions of dollars.

EAB is a prime example of an invasive species - one that is transported outside of its native range and introduced to a non-native ecosystem causing economic or environmental harm. Effective management of invasive species has become a pressing problem because they threaten sustainability by adversely impacting the economy, the environment, and health. Invaders harm biodiversity and degrade the environment [Wilcove et al., 1998]; increase health problems by spreading many diseases [Juliano and Philip Lounibos, 2005], and affect food security by reducing the value of agricultural products [Pejchar and Mooney, 2009]. Due to the substantial impacts on sustainability and human well-being, the international community, including the United Nations' Global Invasive Species Program (GISP), National Invasive Species Council (NISC), and Center for Invasive Species Management (CISM), has called for rapid control of invaders to minimize their adverse impacts [Büyüktahtakın and Haight, 2018].

This study addresses the problem of building a risk-averse spatial-dynamic model to help communities develop cost-effective strategies for the surveillance and control of EAB. Former multi-stage stochastic programming models of EAB surveillance and control decisions [Bushaj et al., 2021b, Kıbış et al., 2021] only considered an expectation criterion in the objective function, the popular Risk Neutral measure. However, due to the intrinsic biological characteristics of the invader and some outside factors, such as careless transportation of infested wood, an infestation could spread fast, and substantial losses of ash trees could happen in a shorter time frame than expected. To alleviate the adverse impacts of experiencing such events, we consider a risk measure in the objective function in addition to the expectation criterion. The incorporation of the risk factors complicates the model, thus requiring advanced computational methodologies to solve it. To tackle the computational difficulty of the proposed complex risk-averse multi-stage stochastic mixed-integer program, we implement the scenario dominance cutting plane algorithm introduced in Büyüktahtakın [2020] to solve the RA-MSS-MIP model more efficiently. The effectiveness of these cuts is studied under the risk-neutral and risk-averse models. We provide insights on how risk-aversion affects decision-making, such as the budget allocated to insecticide treatment and tree removal. We also analyze the benefits of ash trees under risk compared to the original expectation criterion in the risk-neutral problem.

3.2 Literature Review

3.2.1 Risk-Averse Stochastic Programming

Traditional two-stage and multi-stage stochastic programs consider only expectation criterion in the objective function of the optimization problem based on the probability of each scenario, also known as a risk-neutral approach. In problems containing outliers in the distribution of the scenarios, the risk-neutral approach may perform poorly. Assuming that the manager wants to be careful of these extreme scenarios, in the risk-neutral approach, these undesirable outcomes associated with a bad scenario cannot be prevented. The solution obtained from optimizing the expected objective function will perform poorly when one of these outlier scenarios happens. In such cases, risk-averse models become necessary. For example, in a disaster management situation [Escudero et al., 2018a], non-repetitive decisions, such as facility locations [Escudero et al., 2017], may result in a substantial operational cost or even an inability to fulfill the demand for a specific realization of the random parameters [Escudero et al., 2018b].

One of the most popular risk measures is the VaR_α (Value-at-Risk), which represents the maximum possible loss over a time horizon at the confidence level α. While there has been much interest from academic researchers and industry, implementing the VaR is computationally challenging. In the last two decades, a new group of risk measures known as coherent risk measures is originated and studied extensively [Artzner, 1997, Ogryczak and Ruszczyński, 1999]. One of those coherent risk measures is the Conditional Value-at-Risk (CVaR), representing the weighted average of the extreme values in the tail of the distribution beyond the VaR cut-off.

Rockafellar and Uryasev [2000] present a technique on how to evaluate VAR and also optimize CVaR at the same time. It is shown that CVaR can be linearized and easily incorporated into a stochastic optimization problem, making it more preferable to VAR.

Mean-risk models, including VaR, have been widely used in financial optimization; however, the use of mean-risk models with CVaR in stochastic programming models is relatively new [Rockafellar and Uryasev, 2000, Ahmed, 2006, Schultz and Tiedemann, 2006, Miller and Ruszczyński, 2011]. CVaR-based mean-risk stochastic programming has been studied in various applications, such as supply chain management [Alem and Morabito, 2013], reverse logistic network design [Soleimani and Govindan, 2014], and water resources planning [Zhang et al., 2016, Alonso-Ayuso et al., 2018].

The majority of solution approaches presented for risk-averse multi-stage stochastic optimization problems are extensions of the solution techniques proposed for the risk-neutral equivalents Birge and Louveaux [2011]. For example, Shapiro et al. [2013] and Philpott and De Matos [2012] have extended the stochastic dual dynamic programming algorithm to risk-averse problems. Schultz and Tiedemann [2006] develop a Lagrangian decomposition algorithm to solve the scenario-based formulation of two-stage mixed-integer stochastic programming involving CVaR. Zhang et al. [2016] use a nested L-shaped method and investigated multiple cuts to improve the efficiency of a risk-averse multi-stage program. Guo and Ryan [2017] obtain lower bounds using the progressive hedging algorithm to solve time-consistent risk-averse multi-stage stochastic integer programs.

When formulating a risk-averse multi-stage stochastic problem, differently from mean-risk and two-stage stochastic problems, we have to be careful with time consistency. For risk-neutral and two-stage problems, consistency is ensured by default [Gollmer et al., 2011, 2008], but as stages increase, different methods to preserve time consistency are developed [Shapiro, 2012, Pflug and Pichler, 2016]. Time consistency simply states that the decisions taken today must support the decisions that happened yesterday for the scenario that was realized. For example, following the time consistency definition in Alonso-Ayuso et al. [2018] let us assume that the decisions taken up to the realization of a group of scenarios say scenario 1 to scenario n and stage $t' > 1$ (i.e., the decisions from stage 1 to stage $t' - 1$ for this group of scenarios) have been made according to the solution obtained in the original model solved at stage $t = 1$. Then the rationale behind a time consistent risk-averse measure (RAM) is that the solution value to be obtained for any scenario in our group at time stage t' and the later solution values obtained in the scenario tree by the related submodel 'solved' at stage t' should have the same value as in the original model that is solved at stage $t = 1$. In other words, the scenarios that are not in the scenario group we select should not influence the solutions of the submodel at stage t' and later solutions for our scenario group (scenarios 1–n).

Pflug and Pichler [2016] have shown that measuring risk at each time stage separately and measuring the accumulation of risk over a scenario path are inconsistent. Ruszczyński and Shapiro [2006] propose a nested risk measure that proves to be consistent. They ensure its consistency by defining the appropriate conditional risk mappings in each stage, thus, presenting the risk formulation as a recursive function. Homem-de-Mello and Pagnoncelli [2016] propose a similar notion of a nested measure as Expected Conditional Risk Measure (ECRM), stating it to be better from an algorithmic point of view. As the ECRMs consider only continuous variables, another type of risk measure that is increasingly used lately is also the Expected Conditional Stochastic Dominance [Escudero et al., 2018b, 2020], which is based on multi-stage stochastic dominance functional.

3.2.2 EAB Control and Risk-Averse Forest Management Planning

Invasive species pose a serious threat to the ecosystems they invade, and thus much research is performed to design surveillance and control strategies. Management of invasive species is a complex topic as each different invasive species has its specific behavior and biological characteristics. Many optimization problems are proposed for managing invasive species under a limited management budget [Bushaj et al., 2021b, Kıbış et al., 2021, Kıbış and Büyüktahtakın, 2017, Albers et al., 2010, Hof, 1998, Huffaker et al., 1992, Kovacs et al., 2014, Büyüktahtakın et al., 2011, Onal et al., 2020]. For a detailed review of such optimization models, see, e.g., the reviews of Billionnet [2013] and Büyüktahtakın and Haight [2018].

EAB is one of the most damaging invasive species ever to reach the United States. Since its discovery in Michigan in 2002, EAB has spread to more than 37 U.S. states and five Canadian provinces, killing millions of ash trees and costing homeowners and local governments billions of dollars in damages [Aukema et al., 2011]. To slow down the spread of EAB and reduce its harm, city, county, and state planners design surveillance and control strategies, usually with limited budgets. Kıbış et al. [2021] and Bushaj et al. [2021b] addressed the cost-effective allocation of resources to survey and control of EAB. They integrate surveillance and control decisions and jointly optimize them to maximize the benefits of healthy ash trees by saving as many trees as possible. They model dispersal of EAB over time and space similar to discrete reaction-diffusion models (see, e.g., Kıbış and Büyüktahtakın [2019], Büyüktahtakın et al. [2018a]) and surveillance to identify infested trees and their stage of infestation. Modeling EAB dispersal and ash tree health within the optimization model allows for targeted control decisions, such as insecticide treatment and tree removal, which are more cost-effective than naive decisions, such as removing all ash trees without ever surveying the severity of infestation.

The models of Kıbış et al. [2021] and Bushaj et al. [2021b] considered only the expected maximum benefits of healthy ash trees in the objective function without emphasizing the risks and costs of low-chance, high-damage infestation scenarios. Optimal surveillance and control for EAB management may depend on the risk-aversion of managers, who seek to balance between maximizing the expected benefits of ash trees and minimizing the expected damage that could result under the worst possible scenarios of infestation growth.

While accounting for the risk-aversion of decision-makers is standard in finance and economics, such accounting is limited in optimization models for forest management planning, including the surveillance and control of forest invasive species. Among the studies that use risk-averse management and stochastic optimization in forestry operations planning, Alonso-Ayuso et al. [2018] present a time-consistent mean-CVaR multi-stage programming model for planning the harvest of forest land designated for timber production and the construction of access roads needed to transport the timber. Pagnoncelli and Piazza [2017] present a stochastic dynamic programming model for harvest scheduling in which the decision maker wishes to minimize the overall CVaR of her decisions. On the other hand, Eyvindson and Cheng [2016] present a two-stage stochastic programming formulation with CVaR objectives to identify the optimal timing to measure and re-measure forest stands intending to maximize net present value. Yemshanov et al. [2019a] use CVaR in a one-period model to develop optimal surveillance strategies that avoid worst-case outcomes of their surveying actions, where an outcome refers to the expected time to the first detection of a forest pest species.

3.2.3 Key Contributions

The use of risk-averse stochastic programming is limited in forestry operations planning and invasive species management. Former risk-averse stochastic optimization approaches on invasive species control involved only a time domain of a single period. Our approach contributes to the OR and invasive species management literature in the following ways.

Modeling and Algorithmic Contributions. First, we derive a nested risk measure for a maximization problem and integrate it in a scenario-based formulation of a multi-stage stochastic programming problem to obtain a time-consistent formulation. Our time-consistent formulation is different than the node-based formulation of Alonso-Ayuso et al. [2018] in that we define the value-at-risk variable at each stage and under each scenario and include non-anticipativity constraints to impose that decisions and value-at-risk for those scenarios that share the same history up to a certain stage should be the same. Our definition of the risk-related constraints is similar to theirs in that we compute the positive difference between the value-at-risk at each stage and the total benefit up that time stage, and then penalize the expected difference in the objective function. Here, we focus on invasive species surveillance and control, while Alonso-Ayuso et al. [2018] study forest harvest management.

Second, we adapt the scenario dominance cutting planes introduced by Büyüktahtakın [2020] to the case of decision-dependent uncertainty. Specifically, we redefine the scenario dominance concept of Büyüktahtakın [2020] for our problem by considering the endogenous uncertainty modeled in our scenario tree and incorporated the surveillance pattern in defining the scenario dominance sets. We adapt the bounds and cuts to the problem with a maximization objective. Also, different than the method in Büyüktahtakın [2020], we provide a formal cutting plane algorithm, which systematically derives and adds cuts considering the decision-dependent uncertainty and specifics in our problem. While we apply those cuts to solve our case study problem, the proposed scenario dominance framework is general and can be applied to other mean-risk, multi-stage stochastic programming problems.

Third, we perform extensive computational analysis and present results regarding the optimal decision strategies under risk-averse and risk-neutral objectives. Our results demonstrate that scenario dominance cuts reduce the time complexity without an optimality gap. Furthermore, those cuts improve the initial integrality gap, and their performance is not affected by the changes in risk parameters.

Applied Contributions and Policy Insights. To our knowledge, we present the first risk-averse multi-stage stochastic programming model in the invasive species management literature. Multistage stochastic programming is superior to its two-stage counterparts because it can capture the spatial-dynamic features of the EAB infestation and its host trees over multiple time periods. Further, combining the risk-neutral objective with the risk measure allows managers to assess trade-offs between the weighted expectation objective and the risk of loss from low-probability, high-damage EAB scenarios. Using a CVaR risk-averse measure, we improve the benefit for the top $100 \times \alpha\%$ worst-case scenarios compared to the risk-neutral approach. Our model provides several important insights into the spatio-temporal dynamics and risk-averse management of EAB that would not be possible with existing models and methods, as summarized below:

- Our results suggest that, as the manager becomes more risk-averse, insecticide treatment becomes less preferred compared with tree removal, especially for scenarios that involve high infestation spread each year. This is an important practical finding because forest managers often debate over two broad strategies for EAB management: surveillance and insecticide treatment of ash trees versus surveillance and staged removal of ash trees. Our results suggest that the former strategy may provide a higher expected net benefit while the latter strategy may provide a lower risk of outcomes with very low net benefits. The selection of the appropriate strategy will depend on the risk preference of the decision maker.
- Based on our results, increasing risk-aversion by emphasizing the poorly performing scenarios in the objective function might come at a price of reduced expected net benefit. Despite this price, the manager may see this loss as a worthy sacrifice towards the mitigation of possible disaster scenarios.

- Our analysis of surveillance frequency suggests that as we survey less, the expected risk increases due to higher uncertainty about infestation realizations. Thus, a risk-averse manager would want to survey more often.

In Section 3.3, we derive the risk measure for a maximization problem and present the general mean-risk multi-stage stochastic MIP framework. In Section 3.4, we describe notation and formulate the risk-averse multi-stage stochastic mixed-integer programming model for the EAB management in public forests. We derive the scenario dominance cuts and bounds presented in Büyüktahtakın [2020] for a maximization problem and present the associated theoretical results. Finally, in Section 3.6, we describe the implementation details of the model and test six different sets of data estimated by some prior knowledge of the EAB infestation in the state of New Jersey and provide computational results.

3.3 Risk-Averse MSS-MIP Framework

3.3.1 General Formulation of MSS-MIP

Let $\mathcal{T} = \{1, \ldots, T\}$, where T represents the number of stages. Let n^t, h^t, q^t, and m^t represent the number of decision variables, the number of uncertain parameters, the number of integer variables, and the number of constraints, respectively, at time t. We denote the decisions to be taken at each stage $t = 1, \ldots, T$ as $x^t \in \mathbb{R}_+^{n^t - q^t} \times \mathbb{Z}_+^{q^t}$ and the uncertainty observed in stage t as $\xi^t \in \mathbb{R}^{h^t}$, i.e., ξ^t is an $\mathcal{F}^t-$measurable mapping from Ω to $\mathbb{R}^{h^t}$.

The realization of ξ^{t+1} is known after the decision x^t in stage t. We denote the history of the realizations up to state t as $\xi^{[t]} = (\xi^2, \ldots, \xi^t)$ for $t = 2, \ldots, T$ and the decision history up to stage t as $x^{[t]} = (x^1, \ldots, x^t)$ for $t = 1, \ldots, T$. In a t-stage model, the decision-realization sequence is formulated as
$x^1, \xi^2, x^2(x^1, \xi^2), \xi^3, x^3(x^1, x^2, \xi^2, \xi^3), \ldots, x^T(x^{[T]}, \xi^{[T]})$. The general formulation for a T-stage risk-neutral multi-stage stochastic mixed-integer program (MSS-MIP) is written as

$$max \left\{ f^1\left(x^1\right) + \mathbb{E}_{\xi^2}\left[\underset{x^2}{max}\, f^2\left(x^2, \xi^2\right) + \mathbb{E}_{\xi^3|\xi^{[2]}}\left[\ldots + \mathbb{E}_{\xi^T|\xi^{[T-1]}}\left[\underset{x^T}{max}\, f^T\left(x^T, \xi^T\right)) \right]\right]\right]\right\} \tag{3.1}$$

subject to

$$A^1 x^1 \leq b^1 \tag{3.2}$$

$$A^t\left(\xi^{[t-1]}\right) x^{t-1}\left(\xi^{[t-1]}\right) + W^t\left(\xi^{[t]}\right) x^t\left(\xi^{[t]}\right) \leq b^t\left(\xi^{[t]}\right) \quad \forall\, t \in \mathcal{T} \setminus \{1\} \tag{3.3}$$

$$x^1 \in \mathbb{R}_+^{n^1-q^1} \times \mathbb{Z}_+^{q^1}; \quad x^t\left(\xi^{[t]}\right) \in \mathbb{R}_+^{n^t-q^t} \times \mathbb{Z}_+^{q^t} \quad \forall\, t \in \mathcal{T} \setminus \{1\}\,. \tag{3.4}$$

where $A^1 \in \mathbb{R}^{m^1 \times n^1}$ and $b^1 \in \mathbb{R}^{m^1}$ are known; and as time t progresses, the realization of uncertain parameter ξ^t is given by $A^t(\xi) \in \mathbb{R}^{m^t \times n^t}, W^t(\xi) \in \mathbb{R}^{m^t \times n^t}$, and $b^t(\xi) \in \mathbb{R}^{m^t}$; $f^t(x^t, \xi^t) : \mathbb{R}^{n^t \times h^t} \to \mathbb{R}$ represents a linear function for positive integers n^t and h^t such that $f^t(x^t, \xi^t) = c^t(\xi^t)\, x^t(\xi^t)$ where $c^t(\xi^t) \in \mathbb{R}_+^{n^t}$; and $\mathbb{E}_{\xi^t|\xi^{t-1}}[\cdot]$ is the expectation with respect to ξ^t depending on the random realization of the uncertainty, $\xi^{[t]}$.

3.3.2 CVaR, the Risk Averse Measure of Choice

Risk-neutral models simply consider the expected value of the objective function without any estimation of the variability of random outcomes. The expectation objective metric performs poorly under certain scenarios that have outliers with high variability. Depending on whether to maximize or minimize an objective function, one has to account for the large loss of profit or incurred costs falling in the tails of the objective value distribution. Next, we focus on and briefly discuss risk measures that are based on quantiles, such as the Value-at-Risk and the Conditional-Value-at-Risk, to mitigate large losses in the tails of the distribution of objective values.

Definition 3.3.1 *Let $F_X(\cdot)$ represent the cumulative distribution function of a random variable X. The α-quantile of this distribution is called the Value-at-Risk*

(VaR). VaR is represented as:

$$\inf\{\eta \in \mathbb{R} : F_X(\eta) \geq 1 - \alpha\} \tag{3.5}$$

and denoted by $VaR^+_\alpha(X), \alpha \in (0, 1]$.

In a profit maximization context, VaR is the α-quantile of the distribution of the profits providing a lower bound on the distribution of profits, which is not fallen short of with a defined probability $1 - \alpha$. For the profit maximization case, we use the values less than the threshold level of VaR to derive VaR^-_α. Because $VaR^-_\alpha(X) = VaR^+_\alpha(-X)$ the value-at-risk equation for a maximization problem becomes

$$VaR^-_\alpha(X) = -\sup\{\eta \in \mathbb{R} : F_X(\eta) \leq \alpha\}. \tag{3.6}$$

This representation still preserves the monotonicity, translation, and positive homogeneity properties of VaR [Artzner, 1997].

Definition 3.3.2 *Conditional Value-at-Risk (CVaR) for a profit maximization function is defined as*

$$CVaR^-_\alpha(X) = \mathbb{E}(X|X \geq VaR^-_\alpha(X)) \tag{3.7}$$

and represents the conditional expected shortage value, not reaching the value-at-risk at the confidence level α.

CVaR for a random variable X at a confidence level α is formulated as [Rockafellar and Uryasev, 2000]

$$CVaR_{\alpha}^{-}(X) = \inf\left\{-\eta + \tfrac{1}{\alpha}\,\mathbb{E}(\eta - X)_{+}\right\} \tag{3.8}$$

where $(a)_{+} = max\,\{0, a\}$.

Despite the good properties it offers, the VaR is considered a non-coherent risk measure. The VaR also does not consider how bad the scenarios with an objective value below VaR_{α}^{-} can be. On the other hand, the CVaR is coherent and takes into account the scenarios in the α tail, thus, it is preferable to VaR [Acerbi and Tasche, 2002].

3.3.3 Time-Consistent Mean-Risk MSS-MIP

The risk-neutral formulation (3.1)–(3.4) maximizes the expected value function without considering the impact of the extreme and unwanted scenarios on the expectation value. When the objective includes only expectation, the decision-maker will not mainly consider extreme loss scenarios, and a decision that works well for the expectation may result in considerable costs in an extreme scenario. Combining the expected value formulation with risk measures enables the decision-maker to model a trade-off between the profit maximization on average and the risk minimization.

Deriving Time-Consistent CVaR As mentioned above, there are different definitions of time consistency risk measures in literature; we refer to the definition provided in Homem-de-Mello and Pagnoncelli [2016]. Let $(\Xi, \mathcal{F}, P)$ be a probability space and $\mathcal{F}^1 \subset \mathcal{F}^2 \subset \ldots \subset \mathcal{F}^T$ be sub sigma-algebras of $\mathcal{F}$. Let X^t denote a space of $\mathcal{F}^t$ -measurable function form Ξ to $\mathbb{R}$ and $X := X^1 \times X^2 \times \ldots \times X^T$. Then, a multi-stage risk function F is said to be a mapping from X to $\mathbb{R}$.

Homem-de-Mello and Pagnoncelli [2016] define the following multi-period risk function F as expected conditional risk measure (ECRM):

$$F(X^1, \ldots, X^T) = X^1 + \rho^2\left(X^2\right) + \mathbb{E}_{\xi[2]}\left[\rho^3_{\xi[2]}\left(X^3\right)\right] + \ldots + \mathbb{E}_{\xi[T-1]}\left[\rho^T_{\xi[T-1]}\left(X^T\right)\right], \quad (3.9)$$

where $\rho^t_{[\xi^t]}$ represents the risk measure at time t. Homem-de-Mello and Pagnoncelli [2016] prove that any risk function F defined as in (3.9) is time consistent, provided that each $\rho^t_{[\xi^t]}$ satisfies some basic properties that automatically hold, for example, for coherent risk measures.

Using the "tower property" of expectations, F defined in (3.9) could be rewritten as:

$$\begin{aligned} F(X^1, \ldots, X^T) = X^1 + \rho^2\left(X^2\right) + \mathbb{E}_{\xi[2]}\Big[\rho^3_{\xi[2]}\left(X^3\right) + \mathbb{E}_{\xi[3]}\Big[\rho^4_{\xi[3]}\left(X^4\right) + \ldots + \quad (3.10) \\ + \mathbb{E}_{\xi[T-1]}\left[\rho^T_{\xi[T-1]}\left(X^T\right)\right]\ldots\Big]\Big]. \end{aligned}$$

Homem-de-Mello and Pagnoncelli [2016] consider the conditional value at risk of the realization at time t as a particular case of ECRMs defined in (3.9) and (3.10). That is $\rho^t_{[\xi^t]} = CVaR^{[\xi^t]}_{\alpha^t}$, where

$$CVaR^{[\xi^t]}_{\alpha^t} = \max_{\eta^t \in \mathbb{R}} \eta^t - \frac{1}{\alpha^t}\mathbb{E}_{\xi[t]}\left[\left(\eta^t - f^t\left(x^t, \xi^t\right)\right)_+ |\xi^{[t-1]}\right]. \quad (3.11)$$

Using $CVaR^{[\xi^t]}_{\alpha^t}$ as our risk measure in (3.10), we can formulate our 5-stage time-consistent $\mathbb{E}-CVaR$ model as below:

$$\begin{aligned}\max_{x^1,\ldots,x_T} \quad & f^1(x^1) + \max_{\eta^2 \in \mathbb{R}} \eta^2 - \frac{1}{\alpha^2} \mathbb{E}_{\xi^{[2]}} \left[(\eta^2 - f^2(x^2, \xi^2))_+ \right] \\ & + \mathbb{E}_{\xi^{[2]}} \left[\max_{\eta^3 \in \mathbb{R}} \eta^3 - \frac{1}{\alpha^3} \mathbb{E}_{\xi^{[3]}} \left[(\eta^3 - f^3(x^3, \xi^3))_+ | \xi^{[2]} \right] \right. \\ & + \mathbb{E}_{\xi^{[3]}} \left[\max_{\eta^4 \in \mathbb{R}} \eta^4 - \frac{1}{\alpha^4} \mathbb{E}_{\xi^{[4]}} \left[(\eta^4 - f^4(x^4, \xi^4))_+ | \xi^{[3]} \right] \right. \\ & \left. \left. + \mathbb{E}_{\xi^{[4]}} \left[\max_{\eta^5 \in \mathbb{R}} \eta^5 - \frac{1}{\alpha^5} \mathbb{E}_{\xi^{[5]}} \left[(\eta^5 - f^5(x^5, \xi^5))_+ | \xi^{[4]} \right] | \xi^{[4]} \right] | \xi^{[3]} \right] | \xi^{[2]} \right]\end{aligned} \tag{3.12}$$

subject to

$$x^t \in X^t(X^{[t-1]}, \xi^{[t]}) \quad \forall\, t = 1, 2, \ldots, T.$$

where X^t represents the decisions up to stage $t-1$ considering the realization of uncertainty parameter up to stage t ($\xi^{[t]}$).

If the uncertain process is discretized by considering a finite number of realizations of ξ, each random realization over time is named as a scenario, and its index is denoted by ω. Let Ω be the set of scenarios. Each scenario $\omega \in \Omega$ has a corresponding probability p_ω of occurring. Each decision denoted as $x^t_\omega := x^t\left(\xi^t_\omega\right)$, represents the decision for the scenario realization up to stage t, ξ^t_ω, for $t = 1, \ldots, T$. Defining a new variable v^t_ω to account for the positive difference between η^t_ω and x^t_ω for each $t \in \mathcal{T}$, we linearize Equation (3.12) by imposing two additional constraints for each time stage t:

$$v^t_\omega \geq 0 \qquad \text{and} \qquad v^t_\omega \geq \eta^t_\omega - \sum_{t'=1}^{t} f^{t'}(x^{t'}_\omega, \xi^{t'}_\omega) \qquad \forall\, t = 2, \ldots, T. \tag{3.13}$$

For each stage t, the auxiliary variable η^t_ω is a "stage-($t-1$) variable," representing the value-at-risk, $VaR^{[\xi^t]}_{\alpha^t}$. We use an auxiliary variable v^t_ω, which is a "stage-t variable," to represent the stage-t shortage value, not reaching η^t_ω.

The general $\mathbb{E} - CVaR$ optimization problem can be formulated as a dynamic program, which then can be cast into a scenario-based formulation (see Section B.2).

Mean-Risk Scenario-Based MSS-MIP Formulation The CVaR takes into account the benefit for those unwanted scenarios that are below the VaR. However, it does not consider scenarios with a higher benefit than the VaR, which are also included in the expectation objective function. On the other hand, the objective function with only a risk measure, without considering the profit, will provide non-efficient solutions. Consequently, the decision maker prefers a trade-off between risk minimization and benefit maximization. Therefore, in the presence of uncertainty, a widely used approach among practitioners and researchers is to combine the risk measures with the optimization of the expected value of the objective function, leading to the mean-risk models introduced by Markowitz [1952] (see, e.g., Schultz and Tiedemann [2006], Alonso-Ayuso et al. [2018], Ogryczak and Ruszczyński [2001]). Using the $CVaR_\alpha^-$ in Equation (3.8) as a risk measure, we consider the following mean-risk problem within a maximization context:

$$\underset{x \in \mathcal{X}}{max} \left\{ \mathbb{E}(f(x_\omega, \xi_\omega)) + \lambda CVaR_\alpha^-(f(x_\omega, \xi_\omega)) \right\}, \tag{3.14}$$

where λ represents a trade-off coefficient between the expected benefit and the loss due to risk, and $\mathcal{X}$ represents the set of feasible solutions for x. In the mean-risk formulation (3.14), the manager can easily shift from risk-neutral to risk-averse by adjusting the λ parameter. Setting λ to 0, the expression (3.14) would be equivalent to the expectation objective (no matter what α value is). Additionally, giving a certain weight to λ and using α to decide on the size of the tail of the objective value distribution enables the manager not only to decide on the risky values but also to determine how important they are with respect to the expectation.

Using the linearization shown in Equation (3.13), we present the $\mathbb{E} - CVaR$ optimization problem as a general risk-averse multi-stage stochastic mixed-integer program (RA-MSS-MIP) below:

$$P : \max_{\substack{x^1,\ldots,x^T \\ \eta^2,\ldots,\eta^T \\ v^2,\ldots,v^T}} \sum_{\omega\in\Omega} p_\omega \left(\sum_{t=1}^{T} c^t_\omega x^t_\omega + \lambda \sum_{t=2}^{T} \left(\eta^t_\omega - \frac{1}{\alpha^t} v^t_\omega \right) \right) \tag{3.15}$$

subject to

$$v^t_\omega \geq \eta^t_\omega - \sum_{t'=1}^{t} c^{t'}_\omega x^{t'}_\omega \quad \forall\, \omega \in \Omega,\ t \in \mathcal{T} \setminus \{1\}\,, \tag{3.16}$$

$$A^1 x^1_\omega \leq b^1_\omega \quad \forall\, \omega \in \Omega, \tag{3.17}$$

$$A^t_\omega x^{t-1}_\omega + W^t_\omega x^t_\omega \leq b^t_\omega \quad \forall\, \omega \in \Omega,\ t \in \mathcal{T} \setminus \{1\}\,, \tag{3.18}$$

$$x^t_\omega \in \mathbb{R}^{n^t - q^t}_+ \times \mathbb{Z}^{q^t}_+ \quad \forall\, \omega \in \Omega,\ t \in \mathcal{T}, \tag{3.19}$$

$$v^t_\omega \geq 0 \quad \forall\, \omega \in \Omega,\ t \in \mathcal{T}, \tag{3.20}$$

$$x^t_\omega = x^t_{\omega'} \quad v^t_\omega = v^t_{\omega'} \quad \forall\, t \in \mathcal{T};\ \omega, \omega' \in \Omega\ s.t.\ \xi^{[t]}_\omega = \xi^{[t]}_{\omega'}. \tag{3.21}$$

where the random parameters are realized as $\xi^t_\omega = (c^t_\omega, b^t_\omega, A^t_\omega, W^t_\omega)$ for each $t \in \mathcal{T}$ and $\omega \in \Omega$. An important constraint in the scenario-based representation of the RA-MSS-MIP problem in Equations (3.15)-(3.21) is Equation (3.21). The non-anticipativity Equation (3.21) implies that decisions for all scenarios that share the same history up to a specific time t are the same and ensures the implementability of solutions [Wets, 1974, Rockafellar and Wets, 1991]. It also implies that the decision chosen at time t may only depend on the realizations of ξ up to that time period and not on the results of future observations. Furthermore, enforcing non-anticipativity constraints on v^t_ω plays a crucial role in ensuring the time-consistency of the multi-stage decisions.

3.4 RA-MSS-MIP for the Surveillance and Operations Planning of EAB

We apply the mean-CVaR formulation (3.15)-(3.21) to optimize the surveillance and control of the EAB, which has infested large areas covered with ash trees in North America. This section presents the notation and the RA-MSS-MIP formulation for the surveillance, treatment, and removal planning of the ash trees for the EAB infestation. We also provide a verbal description of the model and the assumptions made as well as an example multi-stage scenario tree in Section B.1.

3.4.1 Notation

Sets and Indices

Γ Set of all sites, $\Gamma = \left\{1, 2, \ldots, \bar{\Gamma}\right\}$.

K Set of infestation levels, $K = \{1, 2, 3\}$.

$\mathcal{T}$ Set of time periods, $\mathcal{T} = \{1, \ldots, T\}$.

χ Set of neighboring layers that a spread can happen from a site i; each layer represents a distance-dependent neighbor of site i with similar spread rates.

Ω Set of scenarios in a scenario tree, $\Omega = \left\{1, \ldots, \bar{\Omega}\right\}$.

i Index for site where $i \in \Gamma$.

Θ_i^ι Set of neighboring sites of site i at layer ι.

k Index for infestation level where $k \in K$.

t Index for time period where $t \in \mathcal{T}$.

j Index for neighboring sites of site i at layer ι where $j \in \Theta_i^\iota$.

ω Index for a scenario where $\omega \in \Omega$.

ι Index for neighboring layer where $\iota \in \chi$.

Parameters

p_ω Probability for scenario ω.

c_1 Cost of surveying (inspecting) a tree.

c_2 Cost of treatment.

c_3 Cost of removal.

ζ Monetary value of a susceptible tree.

ϑ_k Penalty value of each infested tree at infestation level k.

r_k Impact rate of each infested tree at infestation level k within site i, i.e., number of new infestations per infested tree at level k.

ϱ Surveillance efficiency, i.e., percent of infested trees that are identified correctly.

τ Discount rate.

δ_t Discount factor at time t, which is equal to $\frac{1}{(1+\tau)^t}$.

Ψ_ω Budget for scenario ω.

θ_k^ι Infestation impact of k^{th}-level infested trees in neighboring layer ι belonging to site j.

κ Maximum number of trees surveyed in each site i.

γ_i Number of surveyed trees in site i under surveillance at time t,i.e., $\gamma_i = min\,(N_{i\omega}^t, \kappa)$.

$p_{j \to i}^\iota$ Probability of infestation spread from site j to i at neighboring layer ι.

$\beta_{k\omega}^t$ Percentage change in belief of infestation after surveillance for infestation level k, at time t, for scenario ω.

$\bar{N}_i$ Initial number of tree population at site i.

$\bar{I}_{ik}$ Initial number of infested tree population at each infestation level k, at site i.

Decision Variables

$N_{i\omega}^t$ Total number of trees at site i, at time t, for scenario ω.

$S_{i\omega}^t$ Number of susceptible trees at site i, at time t, for scenario ω.

$\widetilde{I}_{ik\omega}^t$ Believed number of infested trees at site i, at time t, for infestation level k, for scenario ω before surveillance.

$\dddot{I}_{ik\omega}^t$ Transition number of infested trees at site i, at time t, at infestation level k, for scenario ω after surveillance without considering total tree population.

$I_{ik\omega}^t$ Estimated number of infested trees at site i, at time t, at infestation level k, for scenario ω after surveillance with considering total tree population.

$V_{ik\omega}^t$ Number of treated trees at site i, at time t, at infestation level k, for scenario ω.

$R_{ik\omega}^t$ Number of removed trees at site i, at time t, at infestation level k, for scenario ω.

$H_{i\omega}^t$ Number of trees surveyed at site i, at time t for scenario ω.

$Q_{ik\omega}^t$ Number of infested trees remaining after treatment and removal at site i, at time t, at infestation level k, for scenario ω.

Risk Variables

η_ω^t value-at-risk parameter.

v_ω^t Linearization variable for the risk constraint.

Binary Decision Parameters in Decision Scenario Tree

$$y_\omega^t = \begin{cases} 1 & \text{if surveillance is applied at time } t\text{, for scenario } \omega \\ 0 & \text{otherwise.} \end{cases}$$

Linearization Variables

$$u_{ik\omega}^t = \begin{cases} 1 & \text{if transition population is assigned to infestation level } \; k, \; \text{ at site } i \\ & \text{at time } t \\ 0 & \text{otherwise.} \end{cases}$$

3.4.2 Mathematical Model

Following the convention in Bushaj et al. [2021b], we present the following RA-MSS-MIP formulation for the surveillance and control of the EAB as follows:

$$Max \sum_{\omega \in \Omega} p_\omega \left(\sum_{t \in \mathcal{T}} \left(\delta_t \sum_{i \in \Gamma} \left(\zeta S_{i\omega}^t - \sum_{k=1}^{n} \vartheta_k I_{ik\omega}^t \right) \right) + \lambda \sum_{t=2}^{T} \left(\eta_\omega^t - \frac{1}{\alpha} v_\omega^t \right) \right) \tag{3.22}$$

Subject to :

Risk Linearization Constraint

$$v_\omega^t \geq \quad \eta_\omega^t - \sum_{t'=1}^{t} \delta_{t'} \sum_{i \in \Gamma} \left(\zeta S_{i\omega}^{t'} - \sum_{k=1}^{n} \vartheta_k I_{ik\omega}^{t'} \right) \qquad \forall \; \omega \in \Omega, \; t \in \mathcal{T} \setminus \{1\}\,, \tag{3.23}$$

Initial Total Population

$$N_{i\omega}^{1} = \bar{N}_i \qquad \forall\ \omega,\ i, \tag{3.24}$$

Initial Belief of Infestation

$$\widetilde{I}_{ik\omega}^{1} = \bar{I}_{ik} \qquad \forall\ \omega,\ i,\ k, \tag{3.25}$$

Population Constraint

$$N_{i\omega}^{t+1} = N_{i\omega}^{t} - \sum_{k=1}^{n-2} V_{ik\omega}^{t} - \sum_{k=1}^{n} R_{ik\omega}^{t} \qquad \forall\ \omega,\ i,\ \ t = 1, \tag{3.26}$$

$$N_{i\omega}^{t+1} = N_{i\omega}^{t} - \sum_{k=1}^{n-2} V_{ik\omega}^{t} - \sum_{k=1}^{n} R_{ik\omega}^{t} + \sum_{k=1}^{n-2} V_{ik\omega}^{t-1} \qquad \forall\ \omega,\ i,\ t = 2 \ldots T-1, \tag{3.27}$$

Transition Infestation Level

$$\dddot{I}_{ik\omega}^{t} = \widetilde{I}_{ik\omega}^{t} \left(1 + x_{\omega}^{t} \beta_{k\omega}^{t}\right) \qquad \forall\ \omega,\ i,\ t,\ k, \tag{3.28}$$

Susceptible (Healthy) Tree Population

$$S_{i\omega}^{t} = N_{i\omega}^{t} - \sum_{k=1}^{n} I_{ik\omega}^{t} \qquad \forall\ \ \omega,\ i,\ t, \tag{3.29}$$

Number of Treated and Removed Trees

$$V_{ik\omega}^{t} + R_{ik\omega}^{t} \leq I_{ik\omega}^{t} \sum_{a=max[t-k+1,1]}^{t} y_{\omega}^{a} \qquad \forall\ \ \omega,\ i,\ t,\quad k = 1, \tag{3.30}$$

$$R^t_{ik\omega} \leq I^t_{ik\omega} \sum_{a=max[t-k+1,1]}^{t} y^a_\omega \qquad \forall\ \ \omega,\ i,\ t, \quad k = n-1,\ n, \tag{3.31}$$

Carrying Capacity Constraints

$$I^t_{ik\omega} \leq N^t_{i\omega} - \sum_{d=min(k+1,n)}^{n} I^t_{id\omega} \qquad \forall\ \omega,\ i,\ t,\ k, \tag{3.32}$$

$$I^t_{ik\omega} \leq \dddot{I}^t_{ik\omega} \qquad \forall\ \omega,\ i,\ t,\ k, \tag{3.33}$$

$$\dddot{I}^t_{ik\omega} - I^t_{ik\omega} \leq \bar{N}_i \left(1 - u^t_{ik\omega}\right) \qquad \forall\ \omega,\ i,\ t,\ k, \tag{3.34}$$

$$\left(N^t_{i\omega} - \sum_{d=min(k+1,n)}^{n} I^t_{id\omega}\right) - I^t_{ik\omega} \leq \bar{N}_i u^t_{ik\omega} \qquad \forall\ \omega,\ i,\ t,\ k, \tag{3.35}$$

Believed (Expected) Number of Infested Trees

$$\widetilde{I}^{t+1}_{i1\omega} = \sum_{k=1}^{n} Q^t_{ik\omega} r_k + \sum_{k=1}^{n} \sum_{\iota \in \chi} \sum_{j \in \Theta^\iota_i} Q^{t\iota}_{jk\omega} \theta^\iota_k p^l_{j \to i}\ \forall\ \ \omega,\ i,\ \iota,\ j,\ t = 1 \ldots T-1, \tag{3.36}$$

$$Q^{t\iota}_{ik\omega} = \begin{cases} I^{t\iota}_{ik\omega} - \varrho V^{t\iota}_{ik\omega} - \varrho R^{t\iota}_{ik\omega} & k = 1 \\ I^{t\iota}_{ik\omega} - \varrho R^{t\iota}_{ik\omega} & k = n-1\ \ n \end{cases} \qquad \forall\ \ \omega,\ i,\ t,\ \iota, \tag{3.37}$$

$$\widetilde{I}^{t+1}_{ik\omega} = I^t_{i(k-1)\omega} - \varrho V^t_{i(k-1)\omega} - \varrho R^t_{i(k-1)\omega} \qquad \forall\ \omega,\ i,\ t = 1 \ldots T-1,\ k = 2, \tag{3.38}$$

$$\widetilde{I}^{t+1}_{ik\omega} = \left(I^t_{i(k-1)\omega} - \varrho R^t_{i(k-1)\omega}\right) + \left(I^t_{ik\omega} - \varrho R^t_{ik\omega}\right) \forall\ \omega,\ i,\ t = 1\dots T-1,\ k = n, \tag{3.39}$$

Budget Constraint

$$c_1 \sum_{t\in\mathcal{T}} \sum_{i\in\Gamma} H^t_{i\omega} + c_2 \sum_{t\in\mathcal{T}} \sum_{i\in\Gamma} \sum_{k=1}^{n-2} V^t_{ik\omega} + c_3 \sum_{t\in\mathcal{T}} \sum_{i\in\Gamma} \sum_{k=1}^{n} R^t_{ik\omega} \le \Psi_\omega \qquad \forall\ \omega,\ i,\ t,\ k, \tag{3.40}$$

$$H^t_{i\omega} = \gamma_i x^t_\omega \qquad \forall\ \omega,\ i,\ k,\ t = 1, 2, \dots, T, \tag{3.41}$$

$$\gamma_i = min\left(N^t_{i\omega}, \kappa\right) \quad \forall\ \omega,\ i,\ t, \tag{3.42}$$

Non-anticipativity Constraints

$$\begin{aligned} &N^t_{i\omega} = N^t_{i\omega'} \quad S^t_{i\omega} = S^t_{i\omega'} \quad \widetilde{I}^t_{ik\omega} = \widetilde{I}^t_{ik\omega'} \\ &I^t_{ik\omega} = I^t_{ik\omega'} \quad \dddot{I}^t_{ik\omega} = \dddot{I}^t_{ik\omega'} \quad V^t_{ik\omega} = V^t_{ik\omega'} \ \forall\ i,\ t,\ k, \omega = \omega' \in \Omega\ s.t.\ \xi^{[t]}_\omega = \xi^{[t]}_{\omega'}, \\ &\eta^t_\omega = \eta^t_{\omega'} \quad \upsilon^t_{ik\omega} = \upsilon^t_{ik\omega'} \quad R^t_{ik\omega} = R^t_{ik\omega'} \end{aligned} \tag{3.43}$$

Non-negativity and Binary Restrictions

$$N^t_{i\omega},\ S^t_{i\omega},\ I^t_{ik\omega},\ \widetilde{I}^t_{ik\omega},\ \dddot{I}^t_{ik\omega},\ V^t_{ik\omega},\ R^t_{ik\omega}\ \ge\ 0 \quad u^t_{ik\omega} \in \{0, 1\} \qquad \forall\ \omega,\ i,\ k. \tag{3.44}$$

The objective function of our model shown in Equation (3.22) is a mean-risk scenario formulation of the form shown in Equation 3.15. It allows the manager to adjust the level of risk-averseness by varying the value of λ. As the λ value is increased, the decision-maker becomes more risk-averse by putting a higher weight on the bad scenarios that could occur. Equation (3.23) represents the constraint to linearize the risk parameter.

In Equations (3.24) and (3.25), we define the initial value of the tree population and the expected initial infestation levels for each site i and for all possible scenarios $\omega \in \Omega$ for the initial time period 1. Equations (3.26) and (3.27) keep track of the current population based on decisions made for removing and treating trees. Equation (3.28) computes the believed infestation level for each k. Depending on the surveillance regime and the realization value β, for each infestation level k, the believed number of infested trees in each site i is calculated.

Equation (3.29) keeps track of the number of susceptible trees for each site i by deducting every infested tree from the total population. Equations (3.32)–(3.35) are used as the linearization of the carrying capacity constraint, which is a non-linear equation given below:

$$I^t_{k\omega} = min\left(N^t_{i\omega} - \sum_{d=min(k+1,n)}^{n} I^t_{id\omega}, \ddot{I}^t_{ik\omega} \right) \tag{3.45}$$

Equation (3.45) states that as infestation continues, there is a limit on how many trees can get infested and how many trees can die. In the worst case, if no management decision is taken at all, this limit is constrained by the total number of trees in each site i. Equations (3.32) and (3.33) serve as an upper bound on the number of estimated infested trees, while Equations (3.34) and (3.35) serve as a lower bound by using an auxiliary binary decision variable $u^t_{ik\omega}$.

Equations (3.30) and (3.31) serve as a limit on how many trees can be treated and removed, respectively. Treated and removed trees cannot be more than the number of infested trees. Furthermore, if no surveillance is performed for several periods in a row, then no treatment or removal can be applied.

Equations (3.36) - (3.39) define the number of newly infested trees for each infestation level k for the time period $t + 1$. When trees are neither treated nor removed in infestation level k, they are passed to the next infestation level $k + 1$ in the next period. Once trees reach the last infestation level $k = n$, if not removed, they still stay in that same infestation level. The equations (3.36) - (3.39) represent the growth of infestation over time and space, similar to discrete reaction-diffusion models (see, e.g., Holmes et al. [1994], Kıbış and Büyüktahtakın [2019], Büyüktahtakın et al. [2018a]).

Equation (3.36) estimates the number of trees that will be in infestation level $k = 1$. This is done by collectively calculating the effect of infested trees on susceptible trees nearby. The term $p^{\iota}_{j \to i}$ in Equation (3.36) defines the probability that infestation will spread from site j to site i located at the ι^{th} distance class from j. When calculating dispersal, we cover a radius of 4-km from the infested site. We handle these distance classes using the term θ^{ι}_{i} in Equation (3.36). The unremoved or untreated trees in each of these sites are denoted by the term $Q^{t}_{ik\omega}$ as given in Equation (3.37).

Equation (3.40) ensures that the cost of surveillance, treatment, and removal is under the budget available throughout the planning horizon. Equation (3.43) ensures that scenarios with the same history up to a given stage t share the same decisions until that stage, also known as non-anticipativity constraints in the scenario formulation. Finally, Equation (3.44) defines the non-negativity constraints on the decision variables and defines the linearization variable $u^{t}_{ik\omega}$, as a binary variable. For a more detailed description of the risk-neutral version of the mathematical model above, we refer the reader to the explanation in Bushaj et al. [2021b].

3.5 Scenario Dominance Decomposition

In this section, we present the scenario dominance concept, the scenario sub-problem, the bounds obtained from the scenario sub-problem, and the scenario dominance cuts introduced by Büyüktahtakın [2020] for RA-MSS-MIPs. We first provide those definitions and results for the general risk-averse maximization problem (P) (3.15)-(3.21), and then show the adaptation of these concepts to derive the dominance relations, bounds, and cuts to improve the solvability of our case-study problem in Equations (3.22) – (3.44), which involves decision-dependent uncertainty.

Definition 3.5.1 ***The scenario-ω problem [Büyüktahtakın, 2020]***

The scenario-ω problem P_ω is formulated as follows:

$$Z_\omega = \max p_\omega \left(\sum_{t=1}^{T} c_\omega^t x_\omega^t + \lambda \sum_{t=2}^{T} \left(\eta_\omega^t - \frac{1}{\alpha^t} v_\omega^t \right) \right) \tag{3.46}$$

s.t. Equations (3.16) *to* (3.21)*.*

Remark 1. The scenario-ω problem (3.46) is an MIP, which includes all the variables and the constraints of the original problem P. However, having an objective defined only for a single scenario ω improves the solution time compared to the original problem (3.15)-(3.21).

Definition 3.5.2 ***The relaxed scenario-ω problem***

The relaxed scenario-ω problem P_ω^R is defined as follows:

$$Z_\omega^R = \max p_\omega \left(\sum_{t=1}^{T} c_\omega^t x_\omega^t + \lambda \sum_{t=2}^{T} \left(\eta_\omega^t - \frac{1}{\alpha^t} v_\omega^t \right) \right) \tag{3.47}$$

s.t. Equations (3.16) *to* (3.20)*. Note that P_ω^R is obtained by removing the non-anticipativity constraints* (3.21) *in P_ω.*

Definition 3.5.3 ***The classical scenario-ω problem***

The classical scenario-ω problem P_ω^C is defined as follows:

$$Z^C_\omega = \max p_\omega \left(\sum_{t=1}^{T} c^t_\omega x^t_\omega + \lambda \sum_{t=2}^{T} \left(\eta^t_\omega - \frac{1}{\alpha^t} v^t_\omega \right) \right) \tag{3.48}$$

s.t. Equations (3.16) *to* (3.20) *pertaining only to scenario-ω problem. Not that P^C_ω is obtained by reducing the solution space to that of only scenario-ω.*

Proposition 1 *P^R_ω is a relaxation of P_ω:*

$$Z^R_\omega \geq Z_\omega \quad \forall\ \omega \in \Omega. \tag{3.49}$$

Proof. It is easy to see that the feasible region of P_ω is a subset of the feasible region of P^R_ω. That is $\mathcal{X} \subseteq \mathcal{X}^{\mathcal{R}}$, where $\mathcal{X}$ is the feasible region of the original problem, P, and $\mathcal{X}^{\mathcal{R}}$ is the feasible set of solutions for the relaxed scenario-ω problem, P^R_ω. □

P^C_ω is obtained by removing constraints (3.21) from P and decomposing the model to $|\Omega|$ independent sub-models, each of them related to one scenario ω. On the other hand, P^R_ω includes all the constraints and variables of the original problem except the constraints (3.21). Because P^C_ω and P^R_ω are not defined in the same space, P^C_ω is not a relaxation of P^R_ω. Yet, the optimal objective value of P^C_ω provides an upper bound on the optimal objective value of P^R_ω, as shown in Proposition 2 below.

Proposition 2

$$Z^C_\omega \geq Z^R_\omega \quad \forall\ \omega \in \Omega. \tag{3.50}$$

Proof. Let $(\ddot{x}_\omega, \ddot{\eta}_\omega, \ddot{v}_\omega)$ be the optimal solution for the scenario-ω problem, P^R_ω. Substituting this solution into the objective function of the problem P^C_ω, we have:

$$Z_\omega^C \geq p_\omega \left(\sum_{t=1}^{T} c_\omega^t \ddot{x}_\omega^t + \lambda \sum_{t=2}^{T} \left(\ddot{\eta}_\omega^t - \frac{1}{\alpha^t} \ddot{v}_\omega^t \right) \right) = Z_\omega^R. \tag{3.51}$$

□

Remark 2. From Propositions 1 and 2, it is easy to see that:

$$Z_\omega^C \geq Z_\omega^R \geq Z_\omega \;\; \forall \; \omega \in \Omega. \tag{3.52}$$

3.5.1 Sub-additivity and Upper Bound

Let x^* be the optimal solution for the original problem (3.15)-(3.21) (P), and $Z(x^*)$ be the corresponding objective function value. Let $\dot{x}_\omega$ be the optimal solution for scenario-ω problem P_ω and $Z_\omega(\dot{x}^\omega)$ be the corresponding optimal objective value. Also, let $\bar{x}_\omega$ be the optimal solution to the classical scenario-ω problem P_ω^C and $Z_\omega^C(\bar{x}_\omega)$ be the corresponding objective function value. Next, we provide the classical definition of a sub-additive set function.

Definition 3.5.4 *Let Ω be a set and $\phi : 2^\Omega \rightarrow \mathbb{R}$ be a set function, where 2^Ω denotes the power set. The function ϕ is sub-additive if $\forall \; a, b \subset \Omega$, we have $\phi(a) + \phi(b) \geq \phi(a \cup b)$.*

Definition 3.5.5 ***Multiple Scenario Problem*** *Let $\bar{\Omega} \subseteq \Omega$. Then the multiple scenario problem including the set $\bar{\Omega}$, $P_{\bar{\Omega}}$ is formulated as below:*

$$Z_{\bar{\Omega}} = \max \sum_{\omega \in \bar{\Omega}} p_\omega \left(\sum_{t=1}^{T} c_\omega^t x_\omega^t + \lambda \sum_{t=2}^{T} \left(\eta_\omega^t - \frac{1}{\alpha^t} v_\omega^t \right) \right) \tag{3.53}$$

s.t. Equations (3.16) to (3.21).

Proposition 3 *$Z_{\bar{\Omega}}$ is subadditive on $\bar{\Omega} \subset \Omega$, which means that for $\Omega_1, \Omega_2 \subset \Omega$*

$$Z_{\Omega_1} + Z_{\Omega_2} \geq Z_{\Omega_1 \cup \Omega_2}. \tag{3.54}$$

Proof. Consider P_{Ω_1}, P_{Ω_2} and $P_{\Omega_1 \cup \Omega_2}$. All those three problems share the same feasible region with P. Thus, using the optimal solution of the $P_{\Omega_1 \cup \Omega_2}$, a feasible but sub-optimal solution can be built for each of P_{Ω_1} and P_{Ω_2}. □

Remark 3. Using a similar proof in Proposition 3, it is easy to see that the inequality (3.54) also holds for the classical scenario-ω problem P^C_ω, whose feasible set is only limited to that of only scenario-ω.

Proposition 4 ***(Upper Bound)*** *$\sum\limits_{\omega \in \Omega} Z_\omega$ provides an upper bound on the optimal objective function value of the original problem, i.e.,*

$$\sum_{\omega \in \Omega} Z_\omega \geq Z(x^*). \tag{3.55}$$

Proof. The proof follows from the generalization of the sub-additivity property in Proposition 3 to include all scenarios $\omega \in \Omega$. □

3.5.2 Lower Bound

Let $\dot{x}_\omega$ be the optimal solution for the scenario-ω problem P_ω and $Z(\dot{x}_\omega)$ be the objective value of the original problem P where $\dot{x}_\omega$ is substituted in the original problem objective function.

Remark 4. $Z(\dot{x}_\omega)$ is a lower bound on the objective function value of the original problem as follows:

$$Z(x^*) \geq Z(\dot{x}_\omega) \; \forall \; \omega \; \in \; \Omega. \tag{3.56}$$

As Equation (3.56) holds for each $\omega \in \Omega$, $Z(x^*)$ is bounded below by the maximum of $Z(\dot{x}_\omega)$ over all $\omega \in \Omega$, i.e.,

$$Z(x^*) \geq max_{\omega \in \Omega}\ Z(\dot{x}_\omega). \tag{3.57}$$

3.5.3 Scenario Dominance

As described in Section 3.3.1, we define scenario ξ_ω as the realization path of a random variable ξ over multiple time stages $t \in \mathcal{T}$. Therefore, we derive scenario dominance relations by comparing the realizations of the uncertainty parameter ξ at each time stage t for two different scenarios. Below we first give the original definition of the scenario dominance concept for the general RA-MSS-MIP introduced by Büyüktahtakın [2020] and then provide the definition of scenario dominance for our case-study problem (3.22) – (3.44) presented in Section 3.4.

Definition 3.5.6 ***Scenario Dominance [Büyüktahtakın, 2020].*** *Considering a scenario realization at time $t \in \mathcal{T}$ as $\xi^t_\omega := (c^t_\omega, b^t_\omega, A^t_\omega, W^t_\omega)$ and two specific scenarios ξ_a and ξ_b, scenario ξ_a dominates scenario ξ_b, denoted by $\xi_b \preceq \xi_a$, for the original problem* (3.15)–(3.21) *(P) if*

$$(p_a \geq p_b)\ \wedge\ (c^t_a \geq c^t_b)\ \wedge\ (b^t_a \geq b^t_b)\ \wedge\ (A^t_a \leq A^t_b)\ \wedge\ (W^t_a \leq W^t_b)\ \forall\ t \in \mathcal{T}, a, b \in \Omega$$

and $f(x_\omega, \xi_\omega) = \sum_{\omega \in \Omega} p_\omega \sum_{t=1}^{T} c^t_\omega x^t_\omega$ is a non-decreasing function $\forall t \in \mathcal{T}$ and $\omega \in \Omega$.

Here we define the scenario dominance relations and sets for the EAB surveillance and operations planning problem presented in Section 3.4. We provide a detailed verbal description of the model in Section B.1. An example scenario tree is demonstrated in Figure B.1, and its description is also provided in Section B.1.1 for this problem. This scenario tree depicts the sequences of possible surveillance decisions and the stochastic infestation outcomes over time. There are three outcomes of infestation depending on the surveillance action: low (L) or high (H) if surveillance is performed and medium (M) if surveillance is not performed. Then, L-L-L-M-H represents a specific scenario for a 5-stage problem, where a low realization is observed for the first three years after surveying trees each year, followed by a medium realization without any surveillance, and a high realization at the last stage of the planning horizon after the surveillance is performed.

A dominating scenario is a scenario in which the uncertainty realization at each period is lower, thus providing a higher objective compared to some other scenario. Since our objective is the maximization of the benefits of healthy ash trees, a scenario having a lower EAB infestation provides a higher objective than the one with a higher infestation. For example, defining a scenario having a low (L) infestation realization over five years in consecutive as L-L-L-L-L and a scenario with a high (H) infestation realization over the next five years by H-H-H-H-H, the scenario L-L-L-L-L dominates the scenario H-H-H-H-H if also its probability is larger than the scenario H-H-H-H-H because it leads to less infested trees in each time period.

Similar to the study of Büyüktahtakın [2020], a scenario dominates another scenario if only it yields a higher objective function value. Specific to our problem, we take management action only when we apply surveillance. When we compare two scenarios, we need to consider the surveillance regime of each. For the two scenarios with a different surveillance regime, we cannot be sure which would be the dominating one. We modify the scenario dominance concept defined in Büyüktahtakın [2020] to our problem described in Section 3.4 and present the following definition.

Definition 3.5.7 *In our problem, in addition to the probability of a scenario ω, p_ω, and the left-hand side uncertainty parameter, $\beta^t_{k\omega}$, which represents the percentage change in belief of infestation after surveillance for infestation level k, time t, and scenario ω (see Equation* (3.28)*), we also consider the surveillance regime. Therefore, considering our scenario realization at $t \in \mathcal{T}$ as $\xi^t_\omega := \beta^t_{k\omega}$ for $\omega \in \Omega$ and $k \in K$, and given two scenario realizations ξ_a and ξ_b that share the same surveillance regime, scenario ξ_a dominates scenario ξ_b, denoted as $\xi_b \preceq \xi_a$, for the case-study problem* (3.22) – (3.44)*, if*

$$(p_a \geq p_b) \ \wedge \ (\beta^t_{ka} \leq \beta^t_{kb}) \quad \forall\, t \in \mathcal{T}, \quad k \in K, \ a, b \in \Omega.$$

Based on Definitions 3.5.6 and 3.5.7, we now present the dominance set.

Definition 3.5.8 ***Dominance Set.*** *The set of scenarios which are dominated by scenario $\xi_a \in \Omega \quad (\Lambda^+_{(\xi_a)})$ are described below:*

$$\Lambda^+_{(\xi_a)} = \{b \in \Omega : \xi_b \preceq \xi_a\}.$$

3.5.4 Cuts based on Scenario Dominance

Definition 3.5.9 *Let x^*_ω be the portion of the optimal solution of the original problem that corresponds to scenario ξ_ω and $\bar{Z}(x^*_\omega)$ be the portion of the objective function value at x^*_ω, such that:*

$$\bar{Z}(x^*_\omega) = p_\omega \left(\sum_{t=1}^{T} c^t_\omega x^{t*}_\omega + \lambda \sum_{t=2}^{T} \left(\eta^{t*}_\omega - \frac{1}{\alpha^t} v^{t*}_\omega \right) \right). \tag{3.58}$$

Definition 3.5.10 *Let $\dot{x}_\omega$ be the optimal solution for the scenario ξ_ω problem P_ω and $Z_\omega(\dot{x}_\omega)$ be the corresponding optimal objective function value such that:*

$$Z_\omega(\dot{x}_\omega) = p_\omega \left(\sum_{t=1}^{T} c_\omega^t \dot{x}_\omega^t + \lambda \sum_{t=2}^{T} \left(\dot{\eta}_\omega^t - \frac{1}{\alpha^t} \dot{v}_\omega^t \right) \right). \tag{3.59}$$

Lemma 1 *The optimal objective value of scenario ξ_a problem P_a, $Z_a(\dot{x}_a)$, and the objective value portion of the original problem corresponding to scenario ξ_a, $\bar{Z}(x_a^*)$, are related in the following way:*

$$\bar{Z}(x_a^*) \leq Z_a(\dot{x}_a) \ \forall \ a \in \ \Omega. \tag{3.60}$$

Proof. Assume that for some $a \in \Omega$ we have:

$$Z_a(\dot{x}_a) < \bar{Z}(x_a^*).$$

Then we would have:

$$p_a \left(\sum_{t=1}^{T} c_a^t \dot{x}_a^t + \lambda \sum_{t=2}^{T} \left(\dot{\eta}_a^t - \frac{1}{\alpha^t} \dot{v}_a^t \right) \right) < p_a \left(\sum_{t=1}^{T} c_a^t x_a^{t*} + \lambda \sum_{t=2}^{T} \left(\eta_a^{t*} - \frac{1}{\alpha^t} v_a^{t*} \right) \right). \tag{3.61}$$

Then, we define a new solution to the scenario-ξ_a as below:

$$\dddot{x}_a^t = x^{t*} \ \forall \ t \in \ \mathcal{T}. \tag{3.62}$$

This new solution $\dddot{x}_a^t$ is feasible since x^* is optimal to the original problem and increases the value of $Z_a(\dot{x}_a)$, which contradicts the optimality of $\dot{x}_a$ for the scenario-ξ_a

problem. □

Lemma 2 *Let ξ_a and ξ_b be two scenarios such that $\xi_a \preceq \xi_b$. Let $\ddot{x}_b$ and $\dot{x}_a$ be the optimal solution of scenario ξ_b and ξ_a problems, respectively. Then, the optimal objective value of the scenario ξ_a, $Z_a(\dot{x}_a)$, and the optimal objective value of the scenario ξ_b, $Z_b(\ddot{x}_b)$, have the following relation:*

$$Z_a(\dot{x}_a) \leq Z_b(\ddot{x}_b) \qquad \forall\ a \in \Lambda^{+}_{(\xi_b)}. \tag{3.63}$$

Proof. Let $\tilde{P}_b$ and $\tilde{P}_a$ be two new problems by adding the following inequalities to both scenario-ω problems, in this case, P_b and P_a, respectively:

$$x_b^t = x_a^t\ \forall\ t\ \in \mathcal{T}. \tag{3.64}$$

We define the following two sets as below:

$$\Omega' = \Big(\omega \in \Omega \setminus \{\{a\}, \{b\}\} : \exists\ t \in \mathcal{T}\ such\ that\ \ \xi_\omega^{[t]} = \xi_a^{[t]}\ \ for\ some\ t > 1\Big).$$

$$\Omega'' = \big(\omega \in \Omega \setminus \big\{\Omega' \cup \{a\} \cup \{b\}\big\}\big).$$

Then, we say that $\Omega = \Omega' \cup \Omega'' \cup \{a\} \cup \{b\}$. Note that the set Ω' only includes scenarios $\omega \neq \{a, b\}$ that share a common history with scenario a up to stage t. Therefore, Ω'' is not an empty set, and $\Omega = \Omega' \cup \Omega'' \cup \{a\} \cup \{b\}$. Let $\tilde{x}$ be a feasible solution for both problems, $\tilde{P}_b$ and $\tilde{P}_a$, as below:

$$\tilde{x}_b^t = \ddot{x}_b^t$$

$$\tilde{x}_a^t = \ddot{x}_b^t$$

$$\tilde{x}_\omega^t = \ddot{x}_{\grave{\omega}} \ \forall \ \omega \in \Omega', \ \grave{\omega} \in \Omega$$

$$s.t. \ \xi_\omega^{[t]} = \xi_a^{[t]} \quad for \ t = 1, \ldots, t'$$

$$\xi_{\grave{\omega}}^{[t]} = \xi_b^{[t]} \quad for \ t = 1, \ldots, t'$$

$$\xi_\omega^{[t]} = \xi_{\grave{\omega}}^{[t]} \quad for \ t = t' + 1, \ldots, t$$

$$\tilde{x}_\omega^t = \ddot{x}_\omega^t \quad \forall \ \omega \in \Omega''.$$

The solution defined above is feasible for both of the problems, $\tilde{P}_a$ and $\tilde{P}_b$ because it satisfies all the constraints of the original problem, including the non-anticipativity constraints. We define the optimal objective function values of the problems $\tilde{P}_a$ and $\tilde{P}_b$ as $\tilde{Z}_a$ and $\tilde{Z}_b$, respectively. The solution of $\tilde{x}$ for $\tilde{P}_b$ because for each feasible solution x_b, we have $\tilde{Z}_b(x_b) \geq Z_b(x_b)$ and $\tilde{Z}_b(\tilde{x}) = Z_b(\tilde{x}) = Z_b(\ddot{x}_b)$. Due to no uncertainty in the objective, the decisive parameters are the probability of each scenario to happen p_ω and the uncertainty in left-hand side $\beta_{k\omega}^t$. Because $p_b > p_a$ and $\beta_{kb}^t \leq \beta_{ka}^t$ and a feasible solution vector $\tilde{x}_a$ has to satisfy constraints related to scenario ξ_b due to the Equation (3.64), we get

$$\tilde{Z}_b(\tilde{x}) \geq \tilde{Z}_a(\tilde{x}) \tag{3.65}$$

$$\tilde{Z}_a(\tilde{x}_a) \geq Z_a(\dot{x}_a). \tag{3.66}$$

Because $\tilde{x} = \ddot{x}_b$, we have

$$\tilde{Z}_b(\tilde{x}) = Z_b(\ddot{x}_b). \tag{3.67}$$

Furthermore, we can say that

$$Z_b(\ddot{x}_b) = \tilde{Z}_b(\tilde{x}) \geq \tilde{Z}_a(\tilde{x}) \geq Z_a(\dot{x}_a). \tag{3.68}$$

□

Theorem 1 *Let ξ_a and ξ_b be two scenarios such that $\xi_a \preceq \xi_b$, where $a,\ b\ \in \Omega$ and $a \neq b$. Then, the optimal objective value corresponding to scenario-ξ_b problem, $Z_b(\ddot{x}_b)$, and the objective value portion of the original problem for scenario ξ_a, $\bar{Z}(x_a^*)$, have the following relation:*

$$\bar{Z}(x_a^*) \leq Z_b(\ddot{x}_b) \qquad \forall\ a \in \Lambda^+_{(\xi_b)}, \tag{3.69}$$

where

$$\bar{Z}(x_a^*) = p_a \left(\sum_{t=1}^{T} c_a^t x_a^{t\,*} + \lambda \sum_{t=2}^{T} \left(\eta_a^{t\,*} - \frac{1}{\alpha^t} v_a^{t\,*} \right) \right). \tag{3.70}$$

Proof. From Lemma 1 we have $\bar{Z}(x_a^*) \leq Z_a(\dot{x}_a)$, and from Lemma 2 we have $Z_a(\dot{x}_a) \leq Z_b(\ddot{x}_b)$. Therefore, we can state that:

$$\bar{Z}(x_a^*) \leq Z_a(\dot{x}_a) \leq Z_b(\ddot{x}_b),$$

where $\ddot{x}_b$ is the optimal solution to the scenario-ξ_b problem. □

Remark 5. The trade-off between bound and time. Although scenario-ω formulation P_ω with all the constraints of the original problem provides a better bound than P_ω^C, depending on the size of the problem, P_ω^C could be faster to solve compared to P_ω. Since P_ω^C provides an upper bound on P_ω, the inequality in Equation(3.69) of Theorem 1 can be adapted to the following inequality:

$$\bar{Z}(x_a^*) \leq Z_b^C(\vec{x}_b) \qquad \forall\ a \in \Lambda^+_{(\xi_b)}, \tag{3.71}$$

where $\vec{x}_b$ is the optimal solution for P_b^C.

Proposition 5 ***Strong Scenario Dominance Cuts (ssdc).*** *Let ξ_b be a scenario with $b \in \Omega$, and $\ddot{x}_b^t$ represent the partial optimal solution to the scenario-ξ_b problem and ${x_a^t}^*$ represent the partial optimal solution corresponding to scenario ξ_a, where $a \in \Omega$, in the original problem P, such that $\xi_a \preceq \xi_b$ holds for each time period 1 to t. Then the optimal objective value corresponding to scenario-ξ_b problem over time periods 1 to t, $Z_b^t\left(\ddot{x}_b^t\right)$ and the objective value of the original problem for scenario-ξ_a over time periods 1 to t, $\bar{Z}^t\left({x_a^t}^*\right)$ have the following relation:*

$$\bar{Z}^t\left({x_a^t}^*\right) \leq Z_b^t\left(\ddot{x}_b^t\right) \quad \forall\ a \in \Lambda^+_{\xi_b,t},\ \forall\ t \in\ \mathcal{T}, \tag{3.72}$$

where $\Lambda^+_{\xi_b,t}$ represents the set of scenarios that are dominated by scenario ξ_b for time periods 1 *to* t *and*

$$\bar{Z}^t\left({x_a^t}^*\right) = p_a\left\{\sum_{j=1}^{t} c_a^j {x_a^j}^* + \lambda\sum_{j=2}^{t}\left({\eta_a^j}^* - \frac{1}{\alpha^j}{v_a^j}^*\right)\right\}. \tag{3.73}$$

Remark 6. Adapting sdc to the case-study problem.

The **sdc** (3.71) given in Remark 5 can be adapted to our case-study problem (3.22)

– (3.44) by defining $\bar{Z}(x^*_\omega)$ and $Z^C_\omega(\vec{x}_\omega)$, respectively, as follows:

$$\bar{Z}(x^*_\omega) = p_\omega \left(\sum_{t \in \mathcal{T}} \left(\delta_t \sum_{i \in \Gamma} \left(\zeta S^{t*}_{i\omega} - \sum_{k=1}^{n} \vartheta_k I^{t*}_{ik\omega} \right) \right) + \lambda \sum_{t=2}^{T} \left(\eta^{t*}_\omega - \frac{1}{\alpha} \upsilon^{t*}_\omega \right) \right), \qquad (3.74)$$

$$Z^C_\omega(\vec{x}_\omega) = p_\omega \left(\sum_{t \in \mathcal{T}} \left(\delta_t \sum_{i \in \Gamma} \left(\zeta \vec{S}^{t}_{i\omega} - \sum_{k=1}^{n} \vartheta_k \vec{I}^{t}_{ik\omega} \right) \right) + \lambda \sum_{t=2}^{T} \left(\vec{\eta}^{t}_\omega - \frac{1}{\alpha} \vec{\upsilon}^{t}_\omega \right) \right). \qquad (3.75)$$

Remark 7. Adapting ssdc to the case-study problem.

The **ssdc** (3.72) given in Proposition 5 can be adapted to our case-study problem (3.22) – (3.44) by defining $\bar{Z}^t\left(x^{t*}_\omega\right)$ and $Z^t_\omega(\ddot{x}_\omega)$, respectively, as follows:

$$\bar{Z}^t\left(x^{t*}_\omega\right) = p_\omega \left(\sum_{j=1}^{t} \left(\delta_j \sum_{i \in \Gamma} \left(\zeta S^{j*}_{i\omega} - \sum_{k=1}^{n} \vartheta_k I^{j*}_{ik\omega} \right) \right) + \lambda \sum_{j=2}^{t} \left(\eta^{j*}_\omega - \frac{1}{\alpha} \upsilon^{j*}_\omega \right) \right), \qquad (3.76)$$

$$Z^t_\omega(\ddot{x}^t_\omega) = p_\omega \left(\sum_{j=1}^{t} \left(\delta_j \sum_{i \in \Gamma} \left(\zeta \ddot{S}^{j}_{i\omega} - \sum_{k=1}^{n} \vartheta_k \ddot{I}^{j}_{ik\omega} \right) \right) + \lambda \sum_{j=2}^{t} \left(\ddot{\eta}^{j}_\omega - \frac{1}{\alpha} \ddot{\upsilon}^{j}_\omega \right) \right). \qquad (3.77)$$

Remark 8. In the case of **sdc**, we compare two scenario realizations based on their respective probabilities (p_ω), left-hand side uncertainty parameter ($\beta^t_{k\omega}$), and the surveillance regime. For strong scenario dominance cuts (**ssdc**), we value each of these conditions for up to a time period t. Depending on specific applications, **ssdc** might even cut the optimal solution. In our problem, by involving the surveillance regime in the dominance definition, we prevent these cuts from cutting off the optimal solution.

3.5.5 Cutting-Plane Algorithms

We formally describe the steps of the scenario dominance cutting plane (sdc) algorithm under Algorithm 3.1a and the strong scenario dominance cutting plane (ssdc) algorithm under Algorithm 3.1b. Initially, we formulate the scenario sub-problems. Studying the relation between scenarios and the uncertainty realized after the surveillance, we can define and formulate the dominance relations. In dominance sets, we specify how each scenario relates to another. This part is crucial to decide which cuts are used. For scenario dominance, we group scenarios in five categories considering how many periods in total we surveyed. Using the dominance relations, we can now create dominance sets. Depending on the number of scenarios, a problem may have many cuts. Once solving the randomly selected scenario sub-problem, we use dominated scenarios in set Λ_ω^+ to define the scenario dominance cuts.

To generate the strong scenario dominance cuts, we care that the surveillance perfectly matches in periods from 1 to t for two scenarios compared in the cut. Defining a scenario up to each time t as $\omega^{[t]}$, we define a set of scenarios that are dominated by scenario $\omega^{[t]}$ and denote this set as $\Phi^{+[t]}_{\omega^{[t]}}$. Then, we select the best scenario to solve the sub-problem and add cuts for all the dominated scenarios in set $\Phi^{+[t]}_{\omega^{[t]}}$.

Algorithm 3.1 Algorithms for sdc (3.1a) and ssdc (3.1b)

(a) Scenario Dominance Cut Generation

```
Procedure: Define Dominance Sets
Define: $\beta_\omega$, $p_\omega$, $\Lambda^+_\omega$, $\Lambda^-_\omega$, $N_\omega$
for $\omega \in \Omega$ do
    for $\omega' \in \Omega$ do
        if $\omega$ and $\omega'$ have same number of surveyed periods then
            if $\beta_\omega \leq \beta_{\omega'}$ and $p_\omega \geq p_{\omega'}$ then
                append $\omega'$ to $\Lambda^+_\omega$
                append $\omega$ to $\Lambda^-_{\omega'}$
            else
                append $\omega$ to $N_{\omega'}$
                append $\omega'$ to $N_\omega$
            end if
        end if
    end for
end for

Procedure: Create Cut Map
Define: $\Upsilon$, $n$ {$\Upsilon$ : set of scenarios used as upper bound in the cut; $n$ : cardinality of $\Upsilon$}
Define: $map \ \langle \omega \in \Upsilon, \Lambda^+_s \rangle$    {Mapping a scenario to the scenarios dominated by it}
for $i \in [0, n]$ do
    select a random scenario $\omega^r \in \Lambda^+_{\omega^r}$
    add the pair { $\omega^r : \Lambda^+_{\omega^r}$} to map
end for

Procedure: Add Scenario Dominance Cut
Solve $P_\omega$ and obtain $Z_\omega$
Define $x_{\omega^r}$    {$x_{\omega^r}$: decision variables corresponding to scenario $\omega^r$}
Define $\bar{Z}(x_{\omega^r})$  {$\bar{Z}(x_{\omega^r})$: objective value portion of the original problem for scenario $\omega^r$}
Define cutNum    {$cutNum$ : the number of cuts added $\forall \omega \in \Upsilon$}
for $i \in [0, cutNum]$ do
    select scenario $\omega^r \in \Lambda^+_s$
    add cut: $Z_\omega \geq \bar{Z}(x_{\omega^r})$
end for
```

(b) Strong Scenario Dominance Cut Generation

```
Procedure: Define Strong Dominance Sets
Define: $\omega^{[t]}$   {$\omega^{[t]}$ : scenario $\omega$ from time 1 to $t$}
Define: $\beta_{\omega[t]}$      {$\beta_{\omega[t]}$: uncertainty realization of scenario $\omega$ from time 1 to $t$}
Define: $p_{\omega[t]}$ {$p_{\omega[t]}$: probability of scenario $\omega$ from time 1 to $t$}
Define: $\Phi^{+[t]}_{\omega[t]}$    {$\Phi^{+[t]}_{\omega[t]}$: set of scenarios dominated by scenario $\omega$ from time 1 to $t$}
for $t \in \mathcal{T}$ do
    for $\omega^{[t]} \in \Omega$ and $\omega'^{[t]} \in \Omega$ do
        if $\omega^{[t]}$ and $\omega'^{[t]}$ have the same surveyed periods from time 1 to $t$ then
            if $\beta^{[t]}_\omega \leq \beta_{\omega'[t]}$ and $p_{\omega[t]} \geq p_{\omega'[t]}$ then
                append $\omega'^{[t]}$ to $\Phi^{+[t]}_{\omega[t]}$
            end if
        end if
    end for
end for

Procedure: Add Strong Scenario Dominance Cut
Solve $P_{\omega[t]}$ and obtain $Z^{[t]}_{\omega[t]}$   {$P_{\omega[t]}$: sub-problem for scenario $\omega^{[t]}$, $Z^{[t]}_{\omega[t]}$: objective value of $P_{\omega[t]}$}
Define $x_{\hat{\omega}[t]}$          { $x_{\hat{\omega}[t]}$: decision variables corresponding to scenario $\hat{\omega}$ from time 1 to $t$}
Define $\bar{Z}^{[t]}(x_{\hat{\omega}})$ {$\bar{Z}^{[t]}(x_{\hat{\omega}})$: objective value portion of the original problem for scenario $\hat{\omega}$ from time 1 to $t$}
for $t \in [1, T]$ do
    while $\Phi^{+[t]}_{\omega[t]}$ is not $\emptyset$ do
        select scenario $\hat{\omega}^{[t]} \in \Phi^{+[t]}_{\omega[t]}$
        add cut: $Z^{[t]}_{\omega[t]} \geq \bar{Z}^{[t]}(x_{\hat{\omega}[t]})$
    end while
end for
```

3.6 Computational Experiments

In this section, we present results from our implementation of scenario dominance cuts for solving the case-specific RA-MSS-MIP given in Equations (3.22) - (3.44). We apply the model to sets of data generated for the state of New Jersey.

3.6.1 Implementation Details

We implement each of the models (cpx, sdc, and ssdc) defined above. All of these models are solved using CPLEX v12.7.1 with default settings, and their respective cuts are added as user cuts. We observe results and compare the three models, cpx, sdc, and ssdc, based on solution time, the optimality gap, and the numbers of nodes generated in the branch and bound tree to solve each formulation.

In our multi-stage stochastic problem, uncertainty is observed in the left-hand side parameter $\beta^t_{k\omega}$. For T = 5 stages, we have $3^5 = 243$ scenarios. When defining dominance cuts, we also have to consider the probability of each scenario happening. To calculate probabilities, we use the heuristic algorithm developed in Bushaj et al. [2021b]. Since we have a high number of scenarios, many dominance relations can be defined. In our problem, it is important to define the scenario dominance cuts by considering the surveillance regime. We categorize the surveillance regime into five groups, which are based on how many years we survey during the 5-year period. These categories are defined as: only survey once, survey twice, survey three times, survey four times, and survey every year. The scenario dominance relations are defined by comparing scenarios within these five categories. For each category, we solve the dominating scenario problem, which will provide the best solution among that category, thus in total, we solve five dominating scenario problems. Due to the large number of cuts that can be generated using scenario relations, we randomly select five dominated scenarios from each category. Therefore, in total, we define 25 sdc cuts for the scenario dominance problem, which makes nearly 10% of the total number of scenarios. Regarding the strong dominance cuts, we solve only one dominating scenario problem, but we solve it for each $t = 1, \dots, 5$. The number of strong dominance cuts that can be generated increases as we also consider the time dimension.

Computational experiments were conducted in a Workstation with an 8-core Intel CPU reaching 3.6 GHz and 64 GB RAM running a Windows 10 Enterprise and using CPLEX. We use a time limit of 12,000 CPU seconds for solving each test instance.

3.6.2 Instance Generation and Test Data

We apply our risk-averse model to the case of EAB management in the state of New Jersey. We describe the economic and EAB-related biological and economic data as well as the generation of test instances for the state of New Jersey in Section B.3 and present the results of the computations in the next section.

3.6.3 Results

We solve the data sets generated by using different optimization models with specific features defined as below:

- **cpx**: solving the problem CPLEX 12.7.1 on default settings
- **sdc**: solving the problem with scenario dominance cuts defined on Equation (3.71), which are adapted to our case-study problem (3.22) – (3.44) as described in Remark 6.
- **ssdc**: solving the problem with strong scenario dominance cuts in Equation (3.72), which are adapted to our case-study problem (3.22) – (3.44) as described in Remark 7.

To report computational results, we define the following columns:

- **T**: number of stages;
- **Sce**: number of scenarios;
- **II**: Initial infestation size of the area [small (s) at 1% or large (l) at 2.5% infestation of the total ash trees];
- **DR**: Infestation dispersal rate [slow (s), medium (m), fast (f)];
- **Exp**: Solution approach (cpx, sdc, ssdc) used;
- **Cut**: Number of inequalities added as user cuts;
- **Ctime**: CPU time required to solve the scenario sub-problems and generate scenario dominance cuts (sdc and ssdc);
- **Time**: CPU time required to solve the problem, including Ctime;

- **Tfac**: Time factor improved by cuts over cpx;
- **Node**: Number of nodes explored in the branch and bound tree;
- **Obj**: Best objective value;
- **InitGap**: Percentage integrality gap of formulation before inequalities are added ($InitGap = 100 \times (Obj - relaxObj)/Obj$), where $relaxObj$ and Obj are objective function values of the initial LP relaxation and the best feasible solution by cpx, respectively;
- **GapImp**: Percentage improvement in the integrality gap at the root node ($GapImp = 100 \times (1 - rootObj/relaxObj)$), where $rootObj$ is the objective function value of the LP relaxation after cuts are added at the root node.

Computational Results for Scenario Dominance Cuts In this subsection, we present results regarding the efficiency of sdc and ssdc on six different test files run for five different budget levels as described in Section 3.6.2. Each of these 30 instance combinations is run five times for sdc and ssdc in order to capture the randomness in dominated scenario selection to generate the cuts. Thus, we average over five runs for cpx and 25 runs for sdc and ssdc for each test file, as shown in each row of Table 3.1. This will give an average solution time, which also captures the performance of each model under various budget levels. The Overall Average row gives the results for an overall average of 30 instances. In Table 3.1, the first column noted as (II, DR) identifies the instance by the characteristics it was created. For example, (s,f) stands for the instances generated using a small initial infestation and a fast infestation dispersal rate.

For all instances, we can see that sdc and ssdc cuts reduce the solution time. In addition, they also reduce the number of nodes in the branch and bound tree. As all our instances have five stages, the complexity of the instances increases as the initial infestation is increased, as the infestation spread rate increases, and as the budget tightens. Interestingly, the cut generation time does not increase much as the complexity of the instance increases.

Table 3.1 Experiment Results for CPX, SDC, and SSDC Under Different Infestation Patterns

(II,DR)	Exp	Cut	Ctime	Time	Tfac	Node	Obj	InitGap %	GapImp %
(s,s)	**cpx**	0	0	108	-	0	2,150,592	0.53	-
	sdc	25	8.8	71	1.52	0	2,150,592	0.43	18.7
	ssdc	502	6.8	55	1.96	0	2,150,592	0.34	36.2
(s,m)	**cpx**	0	0	113	-	0	1,192,578	0.78	-
	sdc	25	8.8	76	1.47	0	1,192,578	0.56	28.7
	ssdc	502	7	65	1.72	0	1,192,578	0.45	42.1
(s,f)	**cpx**	0	0	297	-	0	265,106	0.96	-
	sdc	25	8.8	230	1.3	0	265,106	0.8	16.9
	ssdc	502	7.8	168	1.76	0	265,106	0.76	20.6
(l,s)	**cpx**	0	0	108	-	0	420,554	0.61	-
	sdc	25	8.8	80	1.35	0	420,554	0.57	7.5
	ssdc	502	8	75	1.44	0	420,554	0.52	14.9
(l,m)	**cpx**	0	0	615	-	829	196,572	0.97	-
	sdc	25	8.8	326	1.88	169	196,576	0.79	18.4
	ssdc	502	8.2	350	1.76	374	196,576	0.83	13.7
(l,f)	**cpx**	0	0	702	-	1019	-490,969	0.81	-
	sdc	25	8.8	445	1.58	657	-490,969	0.6	26.3
	ssdc	502	8.2	483	1.45	556	-490,969	0.58	29
Overall Average	**cpx**	0	0	324	-	308	622,405	0.78	-
	sdc	25	8.8	205	1.52	138	622,406	0.62	19.8
	ssdc	502	7.7	199	1.69	155	622,406	0.58	25.2

As the overall average shows in Table 3.1, sdc cuts improve the results by a factor of 1.52 while ssdc improves even more, with a factor of 1.69. On average, the cpx solution time is 324 CPU seconds, while the time consumed to generate the sdc and ssdc cuts is 8.8 and 7.7 CPU seconds, respectively. By adding cut generation time to the solution time, sdc and ssdc add up to a total solution time of 205 and 199 CPU seconds, respectively. Furthermore, this improvement is achieved without any loss in the optimal solution. In addition, sdc and ssdc help to improve the solution of the problem's linear programming (LP) relaxation, and thus providing a better initial gap compared to cpx, as shown in the **InitGap** and **GapImp** columns. We refer the reader to Table B.1 in Section B.4 for more information on the LP relaxation solution and the gap improvement with and without sdc and ssdc with respect to cpx.

To investigate the effect of dominance cuts on larger instances, we generate five more test files by increasing the size of the landscape of the base case of 5-by-5, from 6-by-6 (6x6) to 10-by-10 (10x10), each representing a larger landscape than the base case of 5-by-5 test file. For all the larger landscape generations, we use the same initial infestation size (l) and infestation dispersal rate (s). For each unit increase in the landscape size, we increase a unit budget of 5x5 landscape with (l,s), $200,000. For example, for a 6x6 landscape, we have twice the budget of 5x5, for 7x7, we have three times the budget of 5x5, and thus we end up with a six times larger budget than that of 5x5 in the case of a 10x10 landscape.

Table 3.2 presents results for cpx, sdc, and ssdc for larger instances compared to Table 3.1. We have used the same cut generation schema as in Table 3.1, and to account for the change in difficulty, we increase the cut number for **sdc** and **ssdc** models. The tables demonstrate the advantage of the sdc and ssdc cuts over the standard cplex solution for larger instances. Overall, the performance of both sdc and ssdc shows an increasing trend as the instances get larger and harder.

To study the impact of the risk parameters in the time complexity of the model, we perform experiments on a combination of $\alpha = \{0.05, 0.25, 0.5\}$ and $\lambda = \{0.001, 0.1, 1, 10, 1000\}$ values. Computational results show that the sdc and ssdc cuts still improve on risk-averse models for various values of the risk parameters with a similar solution time as the risk-neutral model. For detailed results, see Section B.5.

Table 3.2 Experiment Results for CPX, SDC, and SSDC Under Different Landscape Sizes

(Size)	Exp	Cut	Ctime	Time	Tfac	Binary Variables	Cont Variables	Constraints
5x5	**cpx**	0	0	112	-	91,125	549,180	1,200,020
	sdc	50	8.8	90	1.24	91,125	549,666	1,200,731
	ssdc	716	8	82	1.37	91,125	551,367	1,202,496
6x6*	**cpx**	0	0	369	-	131,220	789,750	1,726,799
	sdc	50	24	270	1.37	131,220	790,236	1,727,510
	ssdc	716	18	230	1.60	131,220	791,937	1,729,2755
7x7*	**cpx**	0	0	503	-	178,605	1,074,060	2,349,356
	sdc	50	66	270	1.86	178,605	1,074,546	2,350,067
	ssdc	716	61	245	2.05	178,605	1,076,247	2,351,832
8x8*	**cpx**	0	0	1,713	-	233,280	1,402,110	3,067,691
	sdc	50	89	883	1.94	233,280	1,402,596	3,068,402
	ssdc	716	75	720	2.38	233,280	1,404,297	3,070,167
9x9*	**cpx**	0	0	2,860	-	295,245	1,773,900	3,881,804
	sdc	50	92	1,772	1.61	295,245	1,774,386	3,882,515
	ssdc	716	85	1,623	1.76	295,245	1,776,087	3,884,280
10x10*	**cpx**	0	0	5,210	-	364,500	2,189,430	4,791,695
	sdc	50	115	1,992	2.62	364,500	2,189,916	4,792,406
	ssdc	716	98	1,719	3.03	364,500	2,191,617	4,794,171
Overall Average	**cpx**	0	0	1,794.5	-	215,662	1,296,405	2,836,228
	sdc	50	65.8	880	1.77	215,662	1,296,891	2,836,939
	ssdc	716	57.5	769.8	2.03	215,662	1,298,592	2,838,704

* *The data used in this test are simulated data, as described in Section 3.6.3.*

Comparison of Risk-Neutral and Risk-Averse Policies We compare the risk-neutral ($\lambda = 0$) model results with three different risk-averse problems. We set $\lambda = 10$ and $\alpha = 0.05$ to represent low risk aversion, $\lambda = 100$ and $\alpha = 0.25$ to represent moderate risk aversion, and $\lambda = 1000$ and $\alpha = 0.5$ to represent high risk aversion (Table 3.3).

Table 3.3 Comparison of Objective Values, Expected Benefit and Risk, and Costs for the Risk-Neutral, Low, Moderate, and High Risk-Averse Models

	Risk-neutral	**Low Risk (α=0.05, λ=10)**	**Moderate Risk (α=0.25, λ=100)**	**High Risk (α=0.5, λ=1000)**
Objective Value[a]	1,310,000	1,406,286	2,517,570	16,507,780
Exp. Benefit ($)	1,310,000	1,309,930	1,309,290	1,301,480
Exp. Risk ($)[b]	-	9,635	12,082	15,206
Exp. Net Benefit ($)	1,121,280	1,121,230	1,121,170	1,120,710
Exp. Treat. Cost ($)	8,956	8,920	8,773	8,632
Exp. Removal Cost ($)	87,045	87,082	87,230	87,373
Benefit of Scn H-H-H-H-H ($)	965,810	965,863	965,978	970,124

[a] *The calculation with units is represented as Expected Benefit ($) + λ * Expected Risk ($)*

[b] *The expected risk does not include preceding λ coefficient*

The optimal objective function values of the risk-averse models are higher than the risk-neutral models because the additional values of risk are added to the objective formulation. When we decompose the objective function into the *expected benefit* $[\mathbb{E}(f(x,\omega))]$ and the *expected risk* $[\lambda CVaR_{\alpha}^{-}(f(x,\omega))]$ in Equation (3.14), we notice that there is a price for being risk-averse – the expected benefit decreases as we increase the risk aversion (α and λ). In addition, as we become more risk-averse a larger subset of risky scenarios is considered, and the expected CVaR value increases. In other words, the VaR value will get smaller, and the expected positive difference between the VaR and the benefits from risky scenarios will get bigger when considering a higher number of risky scenarios. In Table 3.3, the *Objective Value* represents the result of the whole multi-objective formulation shown in Equation (3.22). We also decompose Equation (3.22) into two parts as in Equation (3.14): *Expected Benefit* and *Expected Risk*. The Expected Net Benefit is calculated by reducing the expected total management cost from the expected benefit.

We notice that, as we become more risk-averse, the *Objective Value* increases because of the increase in the *Expected Risk* amplified by the λ coefficient. Decomposing the *Objective Value* into two parts shows that as we are more risk-averse, the *Expected Benefit* decreases, indicating that there is a cost of being risk-averse. The *Expected Net Benefit* decreases, as well. We will analyze the changes in *Expected Net Benefits* with varying risk parameters in more detail in Section 3.6.3. *Expected Treatment* and *Expected Removal* costs, as shown in Table 3.3, help us understand how the optimal decisions change as the manager becomes more risk-averse. The changes in the optimal costs for both treatment and removal under scenario 0, as we become more risk-averse are shown in Table 3.3. Here, we notice a shift in budget allocation between treatment and removal decisions as risk-aversion increases; more of the budget is allocated to removing trees rather than treating trees.

With risk neutrality, the treatment of the asymptomatic infested trees at the earliest stage of infestation provides a higher net benefit than removing those trees or no treatment [Bushaj et al., 2021b]. With risk aversion, the removal of asymptomatic trees is a better option than treatment for the worst-case scenarios. For example, for scenario 0, which involves a high infestation each year, the optimal strategy in the risk-neutral model is to treat as many infested trees as possible for each period within the budget. In the risk-averse model however, a higher benefit is achieved for this scenario by removing these infested trees rather than treating them. Since this scenario has more weight in the risk-averse objective function, the strategy of removing trees rather than treating them increases both the expected benefit and the expected risk portion of the objective function for scenario 0. Further, as the risk aversion increases, the expected treatment cost decreases, and the expected removal cost increases (Table 3.3).

Similar to the benefits case discussed above, the adjustment of the worst-case scenarios under risk aversion also might affect the full objective, expected benefit, and expected net benefit values of other good-performing scenarios.

Figure 3.2 compares the cumulative distribution function of the benefit values for each scenario for the risk neutral and moderate risk-averse models. The risk-averse model now tries to find a better budget allocation strategy between treatment and removal to improve unwanted risky scenarios, even if this improvement can cause a decrease in the expected net benefit value. This is also supported by the shift of the low net benefit values towards higher values (see objectives from 450k to 800k) and the fall in the frequency distribution of the high net benefit values (see objective values from 1,720k to 1,760k) under the moderate risk-averse model compared to its risk-neutral counterpart. We notice that the improvement in low benefit values is obtained to increase the overall expected benefit while we can see the fall of the high net benefit values as the price to pay for being risk-averse.

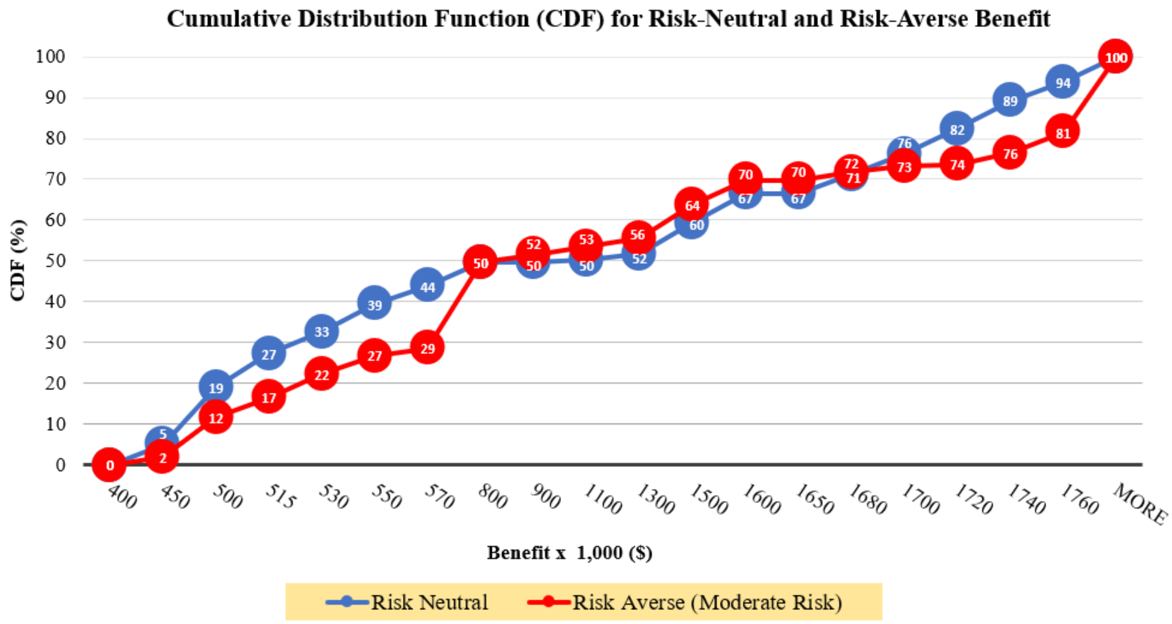


Figure 3.2 The CDF of benefit values of all scenarios for risk-neutral and risk-averse models.

Effect of Risk Parameters in Net Benefits To illustrate further the advantage of using risk measures in our model, we generate some "extreme" scenarios and analyze the *Expected Net Benefit*, the *Expected Profit* over the $100 \times \alpha\%$ worst-case scenarios, and the CVaR under both risk-neutral and risk-averse models. Here, we compute the CVaR of a risk-neutral model by calculating the expected profit over the top $100 \times \alpha\%$ worst-case scenarios. Specifically, "extreme" scenarios are generated by changing the realization of infestation for high and low cases. For the high realization, we increase the severity of the infestation from 40% to 80% more than the manager's expectation, while for the low realization, we decrease the severity from 20% to 40% less than the expectation. We validate the impact of risk in our model by performing experiments with α values of 0.05, 0.1, and 0.2 and for each $\lambda \in \{0, 0.1, 0.5, 1, 5, 10, 50, 100\}$.

Figure 3.3 shows the plot of the *Expected Net Benefit* across a different combination of risk parameters α and λ. Each line represents an α value for a set of λ parameters. The trend we observe is that we can get an increase in *Expected Net Benefit* as we increase λ from 0 to 0.5, and the *Expected Net Benefit* is lower for smaller α values when λ is smaller than 0.5. For the combination of larger α and bigger λ values, emphasizing the worst-case scenarios, the *Expected Net Benefit* also shows a decreasing trend. This happens because when using a bigger α, we improve more of the worst-case scenarios and we emphasize the expected risk with a larger λ.

To investigate further how objective values change under the "extreme" or worst-case scenarios, we plot the expected profit for the top $100 \times \alpha\%$ worst-case scenarios when scenario profit values are sorted in ascending order under both risk-neutral ($\lambda = 0$) and the risk-averse cases ($\lambda > 0$). Figure 3.4 shows how the least benefit scenarios are improved using the CVaR risk measure. Notice that for each of the combinations of the risk parameters α and λ, the expected profit over the top $100 \times \alpha\%$ worst-case scenarios under the risk-neutral case ($\lambda = 0$) is improved.

The lower the α, the higher the expected profit over the top $100 \times \alpha\%$ worst-case scenarios and the corresponding expected profit is increasing as we increase the λ value. A lower α implies that we are more risk-averse, resulting in a higher expected profit over the worst-case scenarios. That being said, after a certain increase in λ, we do not notice further improvement over the $100 \times \alpha\%$ worst-case scenarios for different α parameters.

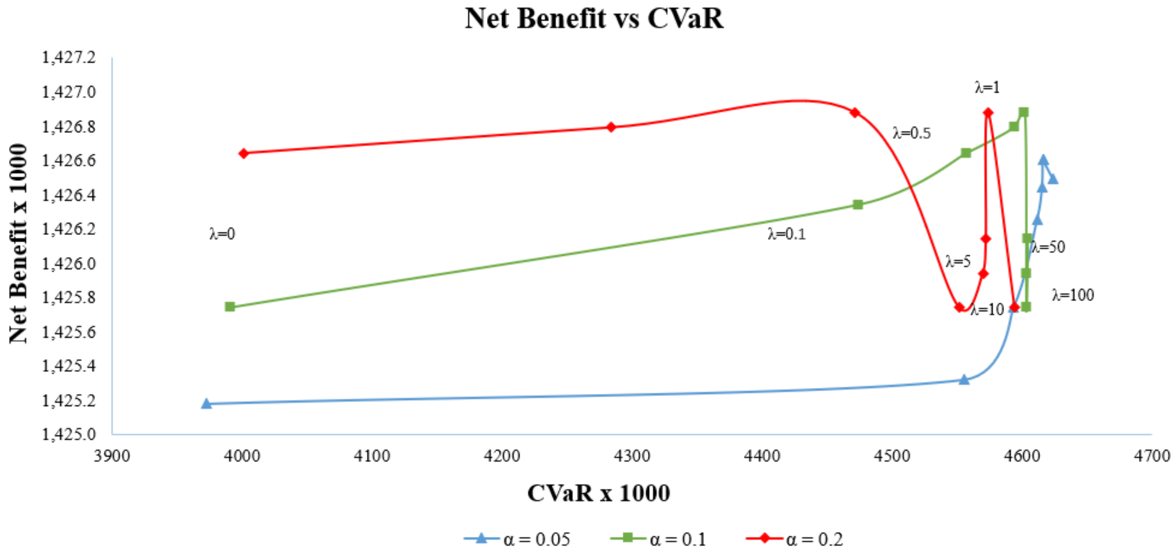


Figure 3.3 Expected net benefit versus CVaR for each combination of risk parameters α and λ under both risk-neutral ($\lambda = 0$) and the risk-averse cases ($\lambda > 0$).

Impact of Risk-Averseness on Surveillance Frequency During our experiments, we notice an interesting relationship between the expected risk and the surveillance frequency. In Table 3.4, we show the risk values for five scenarios having a different surveillance frequency under low, moderate, and high levels of risk aversion. Each scenario is noted using the realization for each time stage. As described in Section 3.5.3, H and L stand for incurring a high and low realization, respectively, while M stands for no surveillance; hence no action is taken in that time stage. For example, the scenario H-H-H-H-H represents surveillance at each period of a 5-stage problem and observing a high outcome at each stage. On the other hand, the scenario

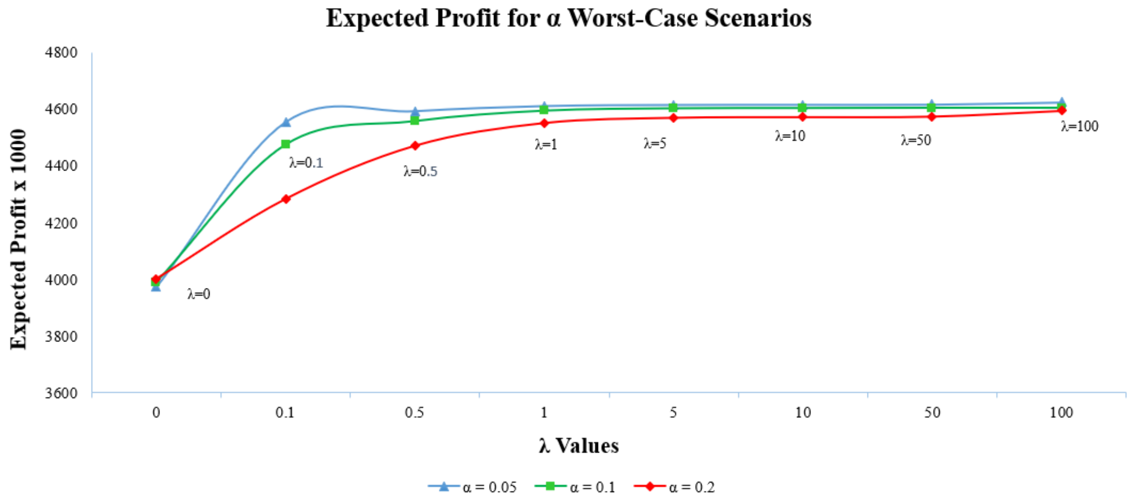


Figure 3.4 Expected profit for the top 100 × α% worst-case scenarios for different values of λ under both risk-neutral ($\lambda = 0$) and the risk-averse cases ($\lambda > 0$).

M-H-H-H-H corresponds to no surveillance in the first stage and thus assuming a medium infestation and doing nothing followed by surveillance in stages two to five and observing a high realization. As can be seen from Table 3.4, as we survey less, the risk increases under each risk-averseness level. This is because as we survey less, we know less about the infestation and make fewer management interventions.

Table 3.4 Risk Values for Five Scenario Realizations with Different Surveillance Frequency Under Various Risk Levels*

Scenario	**Low Risk (α=0.05, λ=10) ($)**	**Moderate Risk (α=0.25, λ=100) ($)**	**High Risk (α=0.5, λ=1000) ($)**
H-H-H-H-H	4,428	12,260	17,752
M-H-H-H-H	6,755	14,351	20,930
M-M-H-H-H	9,650	21,821	36,621
M-M-M-H-H	15,990	30,354	44,150
M-M-M-M-M	20,432	38,421	58,501

* *The risk does not include the λ coefficient*

Budget Constraint Effect on Risk Mitigation Measures To observe the results without the effect of the budget constraint in the risk mitigation, we modify the original model in Equations [(3.22)–(3.44)] by adding the left-hand side of the budget constraint in Equation (3.40) as a negative cost in the objective function in Equation (3.22), thus excluding the budget constraint from the model, as shown below:

$$
\begin{aligned}
Max \sum_{\omega\in\Omega} p_\omega \Bigg(\sum_{t\in\mathcal{T}} \Bigg(\delta_t \sum_{i\in\Gamma} \Bigg(\zeta S_{i\omega}^t - \sum_{k=1}^{n} \vartheta_k I_{ik\omega}^t - c_1 H_{i\omega}^t - c_2 \sum_{k=1}^{n-2} V_{ik\omega}^t - c_3 \sum_{k=1}^{n} R_{ik\omega}^t \Bigg)\Bigg) \\
+\lambda \sum_{t=2}^{T} \left(\eta_\omega^t - \frac{1}{\alpha} \upsilon_\omega^t \right)\Bigg)
\end{aligned}
\tag{3.78}
$$

In this modified model, we also update the risk constraint in Eqution (3.23) to include the cost of surveillance, treatment, and removal as shown in the below constraint:

$$
\begin{aligned}
\upsilon_\omega^t \geq \eta_\omega^t - \delta_t \sum_{i\in\Gamma} \left(\zeta S_{i\omega}^t - \sum_{k=1}^{n} \vartheta_k I_{ik\omega}^t - c_1 H_{i\omega}^t + c_2 \sum_{k=1}^{n-2} V_{ik\omega}^t + c_3 \sum_{k=1}^{n} R_{ik\omega}^t \right) \\
\forall \quad t = 2, \ldots, T, \ \omega \in \Omega
\end{aligned}
\tag{3.79}
$$

The modified model without the budget constraint is obtained by replacing Equations (3.22) and (3.23) with Equations (3.78) and (3.79), respectively, dropping the budget constraint in Equation (3.40) and keeping all other constraints in the original model the same. Table 3.5 presents objective values, expected benefit and risk, and expected costs for varying risk-averseness levels for the modified model without the budget constraint. Similar to the results of the original model shown in Table 3.3, the expected benefit reduces and the expected risk increases as we become more risk-averse. We also observe the same decision shift between treatment and removal costs. As we become more risk-averse, the budget used on treatment decreases while more budget is allocated to the removal actions. Different from the original model with the budget constraint, we notice that a higher amount of money is used for the removal of ash trees. This shows that the budget constraint on the model, apart from the cost of actions we can take, does not affect the trends in the risk-averse decision strategies.

Table 3.5 Comparison of Objective Values, Expected Benefit and Risk, and Costs for the Risk-Neutral, Low, Moderate, and High Risk-Averse Models

	Risk-neutral	**Low Risk (α=0.05, λ=10)**	**Moderate Risk (α=0.25, λ=100)**	**High Risk (α=0.5, λ=1000)**
Objective Value[a]	1,627,967	5,059,190	8,525,780	12,153,700
Exp. Benefit ($)	1,627,967	1,586,430	1,496,830	1,485,580
Exp. Risk ($)[b]	-	3,171,230	3,175,990	3,206,590
Exp. Treat. Cost ($)	7,248.5	7,236.8.2	7,222.3	7,204.8
Exp. Removal Cost ($)	318,691	329,367	329,624	332,310

[a] *The calculation with units is represented as Expected Net Benefit ($) + λ* Expected Risk ($)*

[b] *The expected risk does not include preceding λ coefficient*

3.7 Concluding Remarks

In this research, we developed a risk-averse, multi-stage, stochastic mixed-integer programming model where we incorporate a CVaR risk measure into the objective function to control low-objective scenarios. To facilitate an optimal solution to this complex problem, we defined the scenario dominance sets and generated multiple scenario dominance cuts. We applied the RA-MSS-MIP to the problem of designing surveillance and control strategies for emerald ash borer, a non-native forest insect that damages forests in the eastern U.S. In our application, we tested our dominance cuts and strong dominance cuts and compared solution results to the CPLEX solution in default settings. We also examine how the budget allocation policies change from risk-neutral to risk-averse formulations.

Our results show that our implementation of sdc and ssdc cuts reduces the solution time of risk-averse models by CPLEX by more than 30%. Also, we provide results to show that the performance of the sdc and ssdc cuts does not change much with respect to the variations in risk parameters.

By comparing the risk-neutral and risk-averse policies, we show that there is a price for being prepared for the worst-case scenarios. Despite this price, the manager may see this loss as a worthy sacrifice towards the mitigation of possible disaster scenarios. Our results also imply that as the risk-aversion increases, the budget allocation shifts from inexpensive insecticide treatment to more costly tree removal to slow the infestation. This shift in resources happens because tree removal is more effective at slowing the spread of EAB in scenarios with relatively high rates of spread and damage, and these scenarios are given added weight in the risk-averse management objective. Investigating the effects of risk parameter values on the expected net benefits, we show that the number of poor scenarios included in CVaR measure of risk should be decided carefully by adjusting the (α) parameter.

Here, we study a scenario-based formulation of our multi-stage stochastic MIP problem with ECVaR risk measures. The benefit of the multi-stage formulation is that we can capture the spatial-dynamic features of EAB and its host population of ash trees. Further, the model can address questions of the optimal timing of surveillance and subsequent management decisions under a budget designated for multiple periods larger than two stages. The drawback of the multi-stage formulation is that it cannot handle spatial representation of the survey decision because this would explode the scenario tree. A future study could use a two-stage stochastic programming model to analyze decisions on where to survey in the first stage and subsequent management actions in the second stage. But such a two-stage model sacrifices the ability to analyze the optimal sequence of surveys and account for the spatial dynamics of the pest and its host population.

The uncertainty in our case-study model is exogenous, which is also known as decision-dependent uncertainty because the realization of the uncertain infestation depends on the binary surveillance actions that are integrated into a multi-stage scenario tree [Kıbış et al., 2021]. A future extension of this work could explicitly use surveillance actions in the mathematical formulation and tackle the complexity of the resulting non-linear mixed-integer program. Another future direction of this study could compare the computational performance of our scenario-based RA-MSS-MIP formulation with a node-based multi-stage formulation using Expected Conditional Stochastic Dominance (ECSD) risk measures.

Future research directions include the development of new ways to decide on the selection of cuts for sdc and ssdc. In our problem, we average over a random selection on the dominance sets, but more research could be done to provide insights on which scenarios can perform better when used to derive cuts.

Another possible future direction can be the introduction of a long-distance dispersal mechanism in addition to the 4-km dispersal algorithm. This method could simulate a long dispersal spread of EAB. Finally, we notice from our results that deciding on the values of (α) and (λ) may provide different insights. More work can be done to provide more assistance to the manager by also providing a recommended risk level for the model solved based on the expectations of the management team.

CHAPTER 4

A DEEP REINFORCEMENT LEARNING APPROACH FOR SOLVING MULTI-DIMENSIONAL KNAPSACK PROBLEM

4.1 Introduction

Multidimensional Knapsack Problem (MKP) is an intriguing, strongly NP-Hard problem [Kellerer et al., 2004] with multiple knapsack constraints. MKP can also be considered as a special case of integer programming, restricting the decision variables to 0 or 1. MKP first got attention as a capital budgeting problem [Lorie and Savage, 1955]. MKP is a core resource allocation problem that lies as a sub-problem in many other problems having resource allocation constraints. Thus, contributions to solving MKP can affect a wide range of applications in various businesses, logistics, and computer networks.

Despite many research efforts worldwide and multiple solution approaches to solve the MKP, there is still a large room for improvement, especially for solving large-scale instances efficiently. The recent empowerment of machine learning methods for tackling optimization problems presents a wide area to explore. In particular, reinforcement learning is a promising candidate to outperform current approaches for large MKP instances. Literature suggests that reinforcement and deep reinforcement learning (DRL) approaches can learn solution strategies to solve combinatorial optimization problems [Ma et al., 2019, Bello et al., 2016, Barrett et al., 2020].

In this chapter, we propose a deep reinforcement learning approach combined with a heuristic and a K-Means algorithm to enforce the DRL in a framework to solve large MKP instances. The heuristic reduces the MKP to a more compact representation by evaluating all items and assigning a worthiness value for each. Specifically, we make up an RL environment that arranges a feasible solution according to the worthiness of items suggested by the heuristic. We use this environment as a training ground for an agent where different MKP instances are generated. Then, we propose a K-means clustering iterative algorithm to get a reasonable initial feasible solution that is used to train the DRL algorithm. We construct the DRL framework as a sequential decision-making process where at each step of the algorithm, the agent decides whether the value of a specific item is predicted 1 or 0 until the problem becomes infeasible. So, an episode of the DRL algorithm is made of sequential decisions. Our approach is flexible to train and test using different state-of-the-art DRL models as a sub-method, such as deep Q-learning, policy gradient, or trust region gradient-based methods, in our DRL framework. In our framework, we account for solving MKP instances of various sizes. Our results suggest that we can train RL agents using distinct instances and then generalize prediction to instances of different sizes and distributions. We improve CPLEX performance in terms of solution time with only an average of 0.28% additional solution gap over CPLEX. Furthermore, we offer the option to partially use predicted MKP solutions to find better solutions than those provided by CPLEX.

4.1.1 Notations

$\mathcal{C}$**:** Set of constraints where the respective right-hand side value is appended to each constraint.

$\mathcal{A}$**:** Set of possible actions.

$\mathcal{V}$**:** Set of violated constraints.

Ψ**:** Set of item worth values.

Υ**:** Set of clusters.

υ**:** Index for a cluster where $\upsilon \in \Upsilon$.

$\dot{\upsilon}$**:** Index for a centroid of a cluster υ.

ι**:** Number of K-means iterations.

θ**:** DRL testing steps.

τ**:** DRL training steps.

$\mathcal{Q}$**:** DRL trained model.

Problem Notations:

$\mathcal{J}$ Set of all items, $\mathcal{J} = \{1, 2, \ldots, n\}$.

$\mathcal{I}$ Set of knapsack constraints, $\mathcal{I} = \{1, 2, \ldots, m\}$.

$\mathcal{B}$ Set of knapsack limits, $\mathcal{B} = \{b_1, \ldots, b_m\}$.

$\mathcal{W}$ Set of weights for each item of set $\mathcal{J}$.

$\mathcal{P}$ Set of all P problem instances.

j Index for an item where $j \in \mathcal{J}$.

i Index for knapsack constraint where $i \in \mathcal{I}$.

b_i Index for knapsack limit where $b_i \in \mathcal{B}$.

a_{ij} Index for knapsack weight where $a_{ij} \in \mathcal{W}$.

Binary Decision Parameter

$$x_j = \begin{cases} 1 & \text{if item } j \text{ is selected} \\ 0 & \text{otherwise.} \end{cases}$$

$$\hat{x}_j = \begin{cases} 1 & \text{if item } j \text{ is in reverted item vector is selected} \\ 0 & \text{otherwise.} \end{cases}$$

4.2 Related Work

Early work on MKP treats it as a budget planning problem [Lorie and Savage, 1955, Weingartner, 1966, Weingartner and Ness, 1967]. Applications include but are not limited to computer science [Gavish and Pirkul, 1986, Thesen, 1973], retail business organization [Yang, 2001], and planning and optimization [Shih, 1979]. There exist different solution approaches to solve MKP, such as exact algorithms [Vasquez and Vimont, 2005, Mansini and Speranza, 2012], approximation schema [Nomer et al., 2020], heuristics, and meta-heuristics [Chu and Beasley, 1998, Haul and Voss, 1998, Glover and Kochenberger, 1996, Vasquez et al., 2001, Gaspar et al., 2020]. The main approaches used to get an exact solution of MKP are based on branch-and-bound and branch-and-cut. Different approaches are suggested over the years for branch-and-bound for the MKP (see, e.g., Thesen [1975], Shih [1979], Gavish and Pirkul [1985], Vasquez and Vimont [2005], and Mansini and Speranza [2012]). A number of studies have also used dynamic programming (DP) to get an exact solution Büyüktahtakın [2011]. Among the recent DP attempts are methods proposed by Pisinger [1997], Bertsimas and Demir [2002], and Balev et al. [2008]. Finding an optimal solution is computationally very expensive, motivating researchers to investigate approximation algorithms. Different contributions for polynomial-time approximation solutions are due to Frieze and Clarke [1984] and Caprara et al. [2000]. Again, the computational difficulty of MKP inspired many heuristic algorithms to compute a feasible solution in a reasonable time. We can categorize such heuristics as greedy [Dobson, 1982, Senju and Toyoda, 1968, Loulou and Michaelides, 1979, Yan Yang, 2020], relaxation-based [Bertsimas and Demir, 2002, Hillier, 1969, Balas and Martin, 1980, Magazine and Oguz, 1984], and advanced [Lee and Guignard, 1988, Pirkul, 1987, Fréville and Plateau, 1993]. The greedy approach is based on a relatively simple idea of considering items one by one until the problem becomes infeasible [Fox and Scudder, 1985]. Even more detailed greedy approaches do not fall far from that.

Lately, Deep Reinforcement Learning (DRL) has gained much attention from researchers working in Combinatorial Optimization Problems (COP). Despite many heuristics and exact algorithms proposed to solve COPs, solving large instances is still impossible by the best exact algorithms, such as branch-and-bound [Woeginger, 2003]. Despite the excellent performance in small and medium-sized models, these methods are still not efficient in handling large-scale COPs. Other studies propose different approximation algorithms to deal with the time complexity of these instances [Vazirani, 2013]. Although approximation algorithms improve on the solution time, they frequently involve specific properties that require change each time problem settings are altered.

Recent studies use DRL in Traveling Salesmen Problem (TSP) [Bello et al., 2016, Nazari et al., 2018, Kool et al., 2018], job scheduling [Chen et al., 2017, Li and Hu, 2019], bin packing [Verma et al., 2020, Hu et al., 2017], logistic problems [Pontrandolfo et al., 2002, Delarue et al., 2020] and game playing [Silver et al., 2018, Mnih et al., 2013]. These developments have shown that DRL formulation is suitable for solving sequential decision-making problems. Although many of these applications aim to solve TSP and vehicle routing, it is not difficult to extend these applications to the sequence-to-sequence concept, which is then applicable for solving knapsack problem (KP) instances Bello et al. [2016]. Most of these studies intend to use the power of deep learning towards tackling the curse of dimensionality. Some studies use a Pointer Network architecture [Vinyals et al., 2015, Bello et al., 2016, Gu et al., 2020, Kool et al., 2018]. Others come up with either image or matrix formulations that can represent different classes of COP problems, such as maximum cut, minimum vertex cover, and knapsack [Hubbs et al., 2020, Afshar et al., 2020, Dai et al., 2016, Barrett et al., 2020]. Some more recent studies use DRL to learn from state-of-the-art algorithms and improve them further [Tang et al., 2020, Etheve et al., 2020, Liao et al., 2020, Yang and Rajgopal, 2020].

Vinyals et al. [2015] introduce a Pointer Network architecture, in which the output layer of the deep neural network used in the pointer networks is a function of the input. Bello et al. [2016] use the pointer with reinforcement learning to solve the TSP and knapsack problem. They use a policy gradient with an Advantage Actor-Critic (A3C) algorithm to train their deep neural network. Although initially designed to tackle TSP problems, they report optimal solutions for instances with up to 200 items for the knapsack problem. Gu et al. [2020] present a deep learning algorithm to learn sequential decisions in an unconstrained binary quadratic programming problem (UBQP) since many of the COPs can be generalized into a UBQP. Kool et al. [2018] propose a model based on attention layers with benefits incorporated over the Pointer Network. Using REINFORCE algorithm, they claim to obtain a close-to-optimal solution for two variants of TSP problems with instances up to 100 nodes of the TSP network.

Dai et al. [2017] introduce a new neural network framework for graph-based combinatorial optimization problems. They refer to a *structure2vec*, introduced in Dai et al. [2016], to derive an embedding of the graphs' vertices. They claim that their approach learns effective algorithms for TSP, Maximum Cut, and Minimum Vertex Cover problems. Barrett et al. [2020] use DRL in a different direction as they present the exploratory combinatorial optimization where they aim to improve the agent learning even during test time continuously. Doing so, they claim they can achieve state-of-the-art performance, although further improvements can be made by developing a better starting point. Afshar et al. [2020] propose a DRL approach to solve the knapsack problem. They use state aggregation to extract features and construct states. They compare results with pointer network implementations done in Bello et al. [2016] and Gu et al. [2020] and report that their state-aggregated approach outperforms them. Hubbs et al. [2020] develop a library of reinforcement learning environments consisting of multiple classic optimization problems. Even though they show that DRL is capable of picking up a policy for every problem, it was not able to outperform the heuristic models related to off-line knapsack problems. Kong et al. [2019] follow a slightly different approach to using RL. They investigate whether a theoretically optimal algorithm can be found for online optimization problems. They claim their results are consistent with the behaviors of the optimal algorithms for problems, such as AdWords problem, online knapsack, and secretary.

Another exciting approach is demonstrated in Tang et al. [2020]. Unlike the implementations mentioned above, where the focus is on building a configuration or classical representation of the original models, they propose using DRL to learn state-of-the-art cutting plane methods. They state that their model outperforms human-designed heuristics, and their model can also benefit from the branch-and-cut algorithm. Similarly, Liao et al. [2020] involve DRL in improving global vehicle routing algorithms. Based on their results, they claim they can outperform the A* algorithm, which is a benchmark on a global search. Etheve et al. [2020] show the DRL's strength as they use it to optimize the branching strategy of a Mixed Integer Program (MIP). They present Fitting for Minimizing the SubTree Size (FMSTS), a model that learns the branching strategy from scratch, and they compare it with commercial solvers, such as CPLEX.

To our knowledge, there is no existing DRL approach proposed to solve the multidimensional knapsack problem. Despite the increased complexity due to the multiple items and constraints considered in the multidimensional KP, our DRL method could be generalized to solving other Binary Integer Programming (BIP) problems.

4.2.1 Key Contributions

In this study, we present a new DRL algorithm to tackle the computational difficulty of solving one of the most difficult classes of problems, MKP. Our approach joins the forces of reinforcement learning, K-means, and heuristic approaches and integrates all into a DRL framework to solve COPs, such as the MKP. Because MKP forms the root of many practical problems, Chu and Beasley [1998] state that MKP can be regarded as a general zero-one integer programming with non-negative coefficients. Therefore, improvements on solution methods to this class of problems can be extended to any zero-one integer programming problem. Furthermore, due to a large domain of applications, this NP-hard problem is frequently used as a benchmark problem to compare general-purpose methods in combinatorial optimization [Hanafi and Freville, 1998].

Initially, we present a heuristic that helps us estimate each item's worth. This is a contribution to the heuristic solution approaches, which combines some data analysis and greedy strategy that can be used alone as an algorithm itself. In an attempt to simplify the problem at hand by finding a reasonably starting solution, we propose an unsupervised learning algorithm using K-means to cluster the constraints with a distance-based similarity matrix and relax the problem by only considering a subset of the constraints determined by this K-means algorithm. This approach enables us to reduce the complexity of the problem at hand and get a reasonable feasible solution even for the largest instances to train our DRL algorithm. To harness the power of reinforcement learning, we formulate our problem initially in a 1D environment and then extend it to a 2D environment where an agent is trained to select and deselect items. Together with the above algorithms, a powerful DRL framework is presented to solve large MKP instances. To our knowledge, this is the first DRL model of its kind where a 2D environment is formulated, and an element of the matrix represents an item of the MKP. In addition to the framework, we also create a helper generator for the MKP instances that are randomly generated. We use this generator to produce MKP instances with a different number of items, constraints, and varying distributions of weight and cost parameters.

Among the powerful properties of the DRL framework is its generalization. The environment is set to extract information from different MKP instances and to learn general patterns. The heuristic and K-means algorithm play an important role in generalization as well. For every instance, the heuristic and K-means are executed, enabling a similar pattern to the DRL environment despite the different distribution of the instances. With their help, the solution is concentrated in an isolated area of the 2D environment. Specifically, when we create the DRL environment, we use sorting according to the heuristic, and for all instances, the agent learns “focused” and “advantageous” areas in the environment and where it focuses on searching for an optimal solution. Once trained using the K-means initial solution, the DRL model used with the same training can solve distinct instances. We present results to show that the DRL framework solves instances of different sizes and distributions faster than CPLEX, with a small gap where the DRL agent is only trained once.

4.3 Multi-Dimensional Knapsack Problem Formulation

Here we present the mathematical formulation of the multidimensional knapsack problem. Without loss of generality, we assume all parameters are non-negative. The multidimensional knapsack problem (MKP) is formulated as a binary integer program (BIP) in Equations (4.1) as follows:

$$P \qquad \min \sum_{j=1}^{n} c_j x_j \tag{4.1a}$$

$$\text{s.t.} \sum_{j=1}^{n} a_{ij} x_j \geq b_i \qquad \forall i = 1, 2, \ldots, m. \tag{4.1b}$$

$$x_j \in \{0, 1\} \qquad \forall j = 1, 2, \ldots, n. \tag{4.1c}$$

The objective function (4.1a) minimizes the sum of knapsack (investment) costs over all items $j \in \{1, 2, \ldots, n\}$. Constraints (4.1b) ensure that the total return value of items invested must exceed a given lower bound b_i defined for each constraint $i \in \mathcal{I}$, representing an investment type i. Finally, constraints (4.1c) represent binary integer restrictions on the x_j variables.

4.4 Deep Reinforcement Learning Knapsack Model

To solve our multidimensional knapsack problem, we present a DRL framework to derive a sequential selection policy minimizing the cost without violating any of the original constraints. Originally, our problem P consists of a set of items $\mathcal{J}$, a set of constraints $\mathcal{I}$ with a capacity b_i for each $i \in \mathcal{I}$, a set of weights $a_{ij} \in \mathcal{W}$ for each item $j \in \mathcal{J}$, and constraint $i \in \mathcal{I}$. To solve our problem, we incorporate four different state-of-the-art DRL algorithms, as described in Section 4.4.3.

4.4.1 Heuristic Transformation

To facilitate the multidimensional knapsack problem for the usage in the DRL algorithm we present a heuristic that evaluates the items and sorts them based on their importance, considering their contribution to the objective function and the feasibility of the constraints.

We build our analysis by a worthiness formulation considering the effect of cost values, weights of each item, and the sizes of each knapsack. Despite many different heuristic approaches in literature used to solve knapsack and multidimensional knapsack problems [Boyer et al., 2009, Pirkul], we develop a new heuristic used before the DRL algorithm to improve its performance. In our heuristic, we consider every component of the problem that affects the decision regarding a certain item, such as item cost, item weight for each constraint and, the respective right-hand-side value. We only aim to transform our multidimensional knapsack into a one or two-dimensional vector representation. We do not consider any duality properties or multipliers but only consider proportions between the problem components.

Therefore, we state the following rules to derive our worthiness formulation for each item j:

Rule 1. *The smaller the c_j, the more desirable it is with respect to the objective function.*

Rule 2. *Considering each constraint/knapsack alone, then the larger the a_{ij}, the more valuable the item is in terms of satisfying the knapsack constraint i.*

Rule 3. *The smaller the b_i, the easier it is to satisfy constraint i with the selected items.*

Going further in our analysis, we consider the relationship between each of the statements above, and we provide the following extensions of the rules:

Rule 4. *Regarding Rules 2 and 3, the higher the $\frac{a_{ij}}{b_i}$, the more desirable item j is because it fills a larger portion of the knapsack.*

Rule 5. *Using Rules 1 and 4, the lower $\frac{c_j}{a_{ij}/b_i}$, the more valuable item j is since we prefer an item j with a lower numerator and a larger denominator.*

Importance of an item: Based on the above rules, we formulate the worthiness of an item j for a multi-dimensional knapsack formulation as:

$$r_j = \frac{\sum_{i=1}^{|\mathcal{I}|} \frac{c_j}{a_{ij}/b_i}}{|\mathcal{I}|} \qquad \forall\ j = 1, 2, \ldots, n \tag{4.2}$$

The lower the worthiness score of an item calculated in Equation (4.2) is, the more beneficial we see it to be selected. Therefore, sorting the items based on this ratio in ascending order gives us a vector where the most favorable items are listed first.

Throughout the DRL algorithm, we use a reverted vector of items based on these ratios. We denote the items in the original vector as $x_j\ \forall j \in \mathcal{J}$ and the items in the reverted vector as $\hat{x}_j\ \forall j \in \mathcal{J}$. For example, in Figure 4.1, we move from the regular vector in blue to the new order in green based on the worthiness ratios. We use some helper methods to keep a relationship between both representations that allow us to easily switch from one to another.

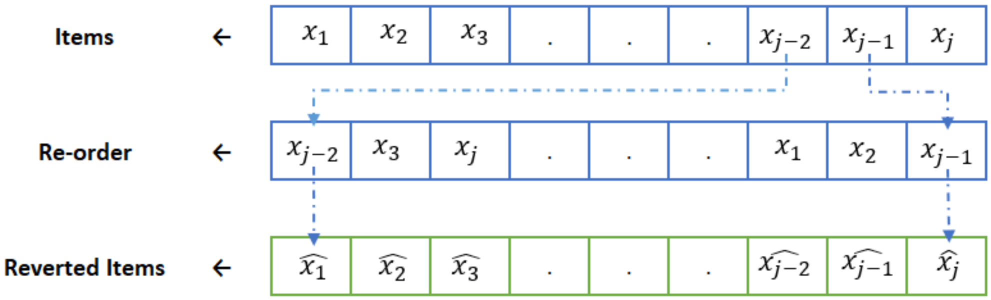


Figure 4.1 Reverting the items based on the worthiness ratios.

Algorithm 1 shows the step-by-step procedures of the heuristic algorithm. Initially, we calculate the item worth for each item in the knapsack and store them in a vector. Then on a second procedure, we sort them in ascending order, but we maintain their original position. We use some utility methods to convert from the sorted items to the original problem.

Algorithm 1 Knapsack Transformation Heuristic

```
Procedure: Calculate Item Worth
Input: $c_j, a_{ij}, b_i$                  {Item cost, item weight, and constraint $i$ right-hand side.}
Output: $\Psi$                                                            {The item worth set.}
for $j \in \mathcal{J}$ do {for each item}
    for $i \in \mathcal{I}$ do {for each knapsack}
        $r_j = +\frac{c_j}{a_{ij}/b_i}$                                  {Calculate worthiness ratio.}
    end for
    append $\frac{r_j}{|\mathcal{I}|}$ to $\Psi$
end for

Procedure: Sort Elements According to Item Worth
Input: $\Psi$                                                                {Item worth list.}
Output: $S$, $S'$ {$S$ - sorted values, $S'$ keeps re-ordered indices for items according to worthiness ratio.}
$S, S' \longleftarrow$ sortAscending($\Psi$)
```

4.4.2 K-means Algorithm and Initial Solution

To provide a feasible good starting solution for the DRL algorithm and a benchmark of how a solution should look like as an input into the RL training algorithm, we use the K-means algorithm, which divides the constraints of the knapsack instance into multiple clusters and generates an initial solution. Algorithm 2 shows each step of the K-means Constraint Clustering Algorithm in detail.

We provide all constraints (weights and right-hand side) as an input for the K-means algorithm. We calculate the similarity of two constraints using an $n+1$-dimensional Euclidean distance between two $n+1-$dimensional vectors. Each of the vectors contain n items and the right-hand side value of the corresponding constraint. Considering two vectors $d_f = [a_{f1}, \ldots, a_{fn}, b_f]$ and $d_k = [a_{k1}, \ldots, a_{kn}, b_k]$ where f and $k \in \mathcal{I}$, we calculate each distance between d_f and d_k, D_{d_f,d_k} as:

$$D_{d_f,d_k}{}^2 = \sum_{j=1}^{n+1} \left(d_{fj} - d_{kj}\right)^2, \tag{4.3}$$

where d_{ij} is the j^{th} element of the constraint vector d_i for $i \in \mathcal{I}$. At the starting point, we assign a centroid for each cluster by selecting one of the constraints randomly for each cluster. A distance map is populated by calculating the distance between each vector d_i for $i \in \mathcal{I}$ and the centroid vectors $\upsilon_k \in \Upsilon$, using Equation (4.3). Using this distance map, we reassign the vector d_i to the closest centroid or cluster. With the new cluster assignment, we recalculate cluster means and new centroids. We repeat this procedure for a pre-set number of iterations. We limit the number of iterations since we do not want to spend too much time on the K-means algorithm. Here, we only need a good feasible solution to design our RL environment.

Algorithm 2 K-means Constraint Clustering

```
Procedure: Generate Clusters
Input: $\iota, \Upsilon, \mathcal{C} = \mathcal{I} \cup \mathcal{B}$  {Number of iterations, set of clusters, set of constraint vectors for problem $P$.}
$\dot{\upsilon}_1 = \zeta_1, \dot{\upsilon}_2 = \zeta_2, \ldots, \dot{\upsilon}_{|\Upsilon|} = \zeta_{|\Upsilon|}$ {Assign first constraints as centroids for each cluster $\upsilon \in \Upsilon$, where $\dot{\upsilon}$ is a centroid of a cluster $\upsilon$ and $\zeta \in \mathcal{C}$}
Output: $\upsilon_1, \upsilon_2, \ldots, \upsilon_{|\Upsilon|}$  {Output $|\Upsilon|$ clusters.}
for each $\iota$ do
    for $\zeta \in \mathcal{C}$ do
        for $\dot{\upsilon} \in \Upsilon$ do
            $D_{\zeta,\dot{\upsilon}} = \sqrt{\sum_{j=1}^{n+1} (\zeta - \dot{\upsilon})^2}$
        end for
    end for
    for $\zeta \in \mathcal{C}$ do
        Assign $\zeta$ to the closest cluster $\upsilon \in \Upsilon$  {Reassign constraints to the closest cluster $\upsilon$ with centroid $\dot{\upsilon}$.}
    end for
    for $\upsilon \in \Upsilon$ do {for each cluster}
        for $\zeta \in \upsilon$ do
            $\dot{\upsilon} = \left[\frac{\sum \zeta_1}{|\upsilon|}, \ldots, \frac{\sum \zeta_{|\upsilon|}}{|\upsilon|}\right]$ {Recalculate the mean distance for each cluster dimension and assign it as the new centroid.}
        end for
    end for
end for

Procedure: Cplex Recursive Selection
Input: $\upsilon_1, \upsilon_2, \ldots, \upsilon_{|\Upsilon|}$
Define: $\bar{P}$  {$\bar{P}$ is the unconstrained original problem.}
Define: $\mathcal{V}$  {Set of violated constraints.}
Output: $\bar{X}$, $\bar{Z}$  {Feasible solution and objective value of $\bar{P}$.}
for $\upsilon \in \Upsilon$ do
    Append the farthest pair of vectors of $\upsilon$ to $\bar{P}$
end for
Solve $\bar{P} \longmapsto \mathcal{V}$  {Solve $\bar{P}$ and determine the set of the violated constraints $\mathcal{V}$.}
while $\mathcal{V}$ is not $\emptyset$ do
    if $|\mathcal{V}| > 10$ then
        Append five most violated constraints to $\bar{P}$
    else
        Append all the remaining constraints to $\bar{P}$
    end if
    Solve $\bar{P} \longmapsto \mathcal{V}$  {Recalculate violation set $\mathcal{V}$.}
end while
```

4.4.3 DRL Model

In recent years, different algorithmic approaches using neural network approximators have been proposed for RL to tackle large problems, especially COPs. Among the most popular ones are Deep Q-learning, policy gradients methods, and trust region gradient methods. For each algorithm, we serve the state and possible actions as input, and we get back an action as an output. We perform training and testing using four state-of-the-art algorithms. For example, we train the model and test it using the Advantage Actor-Critic (A2C) method introduced in Mnih et al. [2016].

The authors propose a DRL framework that uses asynchronous gradient descent to optimize deep neural network controllers. They use parallel actor-learners to update a shared model instead of experience replay used in Deep Q Network (DQN) to achieve a stable learning process. We also train and test our knapsack framework using a Double DQN with an experience replay presented in Schaul et al. [2015]. Although using too much memory and computational power, experience replay can decorrelate episode updates. Among the trust region policy optimization algorithms, we also test our results in Actor-Critic using Kronecker-Factored Trust Region (ACKTR) developed by Wu et al. [2017]. ACKTR is a scalable trust-region optimization algorithm for actor-critic methods. The authors use a Kronecker-factored approximation to natural policy gradient allowing the covariance matrix of the gradient to be inverted efficiently. Their paper is among the first to try to combine the benefits of different groups of algorithms (trust region policy optimization and the policy gradient). Another such algorithm is proposed by Schulman et al. [2017] to attain the data efficiency and reliable performance of trust region policy optimization while only using the first-order optimization.

4.4.4 One-dimensional Knapsack Environment

In this section, we describe a one-dimensional vector representation of potential solutions of the multidimensional knapsack problem. We represent the states of the DRL algorithm as the combination of all possible binary selections in the vector where each element of the vector represents an item. Figure 4.2 shows the environment used to describe the one-dimensional formulation achieved using our heuristic in Algorithm 1. Our heuristic provides the order of the items in a vector based on their importance, and when sorting it, the agents will learn to concentrate on searching for the solution in a slightly isolated area. For example, when we sort the items according to the item's worth, the first items are more likely to be selected, and the last items are more likely to be ignored. This approach creates a focus area of decisions. We aim to train the agent to learn "swimming" in that area. Below, we describe episodes, state, action, and a reward function in our deep reinforcement learning algorithm using this 1D knapsack environment.

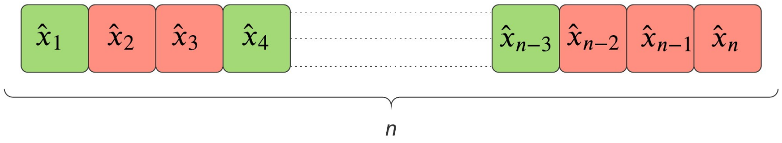


Figure 4.2 One-dimensional vector representation of our problem.

Episode: We define an episode as the steps taken from a current state until we find an infeasible solution or the maximum number of steps per episode is reached. In each episode, we aim to maximize the average reward.

State $s(P)$: Each state describes the current selection of an item $j \in \mathcal{J}$, as either 0 or 1. The heuristic defined in Algorithm 1 simplifies our states from the collective combination of items with their costs and weights and knapsack constraints to a simple state-space of 2^n where n represents the number of items, and each item can take a value of 0 or 1. Thus, the state space is only defined by the selection of items free from the structure of constraints. We denote a state of the problem as $s(P)$.

Actions: We allow the agent to select/deselect an item at a step of each episode based on the current state. Therefore, we have n potential actions where n is the number of items at each step of the algorithm. We denote each action as A_j where j denotes a specific item. For each step, our algorithms are fed a certain state describing whether an item j is selected or deselected, and an action is taken upon that state. For example, if action A_j is taken at state $s(P)$, then if item j was selected, we deselect it, or if an item was not selected, we select it. This will form a new state

$s'(P)$. At each step, we only take one action A_j for a particular item j.

Reward Function: Our reward function is guided using the original problem. Since we minimize the objective function, we aim to reduce the objective value without violating any constraints. Therefore, we give rewards based on four different situations. First, if the action reduces the objective value and leads to another feasible state, the agent is given a positive reward. Second, if the objective is increased and the problem remains feasible, then a small negative reward is given. Third, if an action leads to infeasibility, then a high negative reward is given. Lastly, if an action leads to a better solution than the starting solution, we give a higher positive reward, and if the solution is still feasible, we continue the next step. Let Z_s be the objective value of a certain state s and j be the item currently selected in a step. We then can formulate the reward function as:

$$r(s(P), A_j) = \begin{cases} +Z_s & \text{if a better solution than the starting solution is found} \\ +c_j & \text{if the objective is reduced and feasibility is maintained} \\ -c_j & \text{if the objective is increased and feasibility is maintained} \\ -Z_s & \text{otherwise} \end{cases} \tag{4.4}$$

Based on our results, there are some disadvantages of using the 1D representation of the model. First, as instances' size increases, the number of states and actions increase exponentially, thus, having a considerable effect on training time. Therefore, the model training time and the time agent needs to learn will also increase. To overcome these weak points, we reformulate our model using a two-dimensional representation, gaining more advantage of the heuristic and lower action space, as presented in the next section.

4.4.5 Two-dimensional Knapsack Environment

Here we develop a novel two-dimensional knapsack environment and formulation to overcome the weaknesses of the one-dimensional representation described in Section 4.4.4. The heuristic is used in this representation as well, but besides, we reshape a vector of n items into a square two-dimensional matrix of $\sqrt{n}$, as shown in Figure 4.3.

The reshape from the 1D representation is done after using the sorting heuristic. Here, the items are sorted according to their worthiness, starting from the top-left first cell to the bottom-right the last cell. Similar to the isolated area of the solution on the 1D representation, sorting and locating the items based on their worthiness in the 2D matrix will help the agent learn faster to select or deselect items.

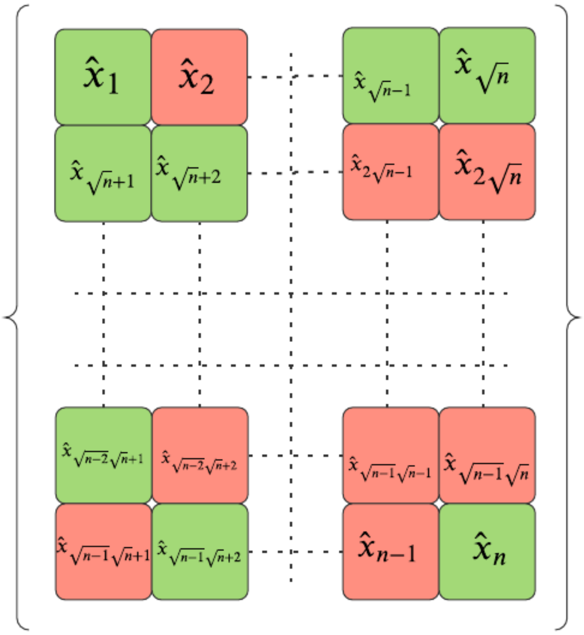

$\hat{x}_1$
$\hat{x}_2$
$\hat{x}_{\sqrt{n}-1}$
$\hat{x}_{\sqrt{n}}$
$\hat{x}_{\sqrt{n}+1}$
$\hat{x}_{\sqrt{n}+2}$
$\hat{x}_{2\sqrt{n}-1}$
$\hat{x}_{2\sqrt{n}}$
$\hat{x}_{\sqrt{n-2}\sqrt{n}+1}$
$\hat{x}_{\sqrt{n-2}\sqrt{n}+2}$
$\hat{x}_{\sqrt{n-1}\sqrt{n}-1}$
$\hat{x}_{\sqrt{n-1}\sqrt{n}}$
$\hat{x}_{\sqrt{n-1}\sqrt{n}+1}$
$\hat{x}_{\sqrt{n-1}\sqrt{n}+2}$
$\hat{x}_{n-1}$
$\hat{x}_n$

Figure 4.3 Two-dimensional matrix representation of the state space.

As an extension of the 1D formulation, some of the design properties are inherited. Changing the shape of the environment does not change our state space.

Instead, it reduces the action space and also provides a path-like movement into selecting and deselecting items.

Episode: Similar to the 1D formulation, the episode starts with a feasible solution and moves on the 2D matrix until the solution (state) becomes infeasible or the maximum number of steps for each episode is reached.

States s: Changing the dimensions of the representation does not change our state space. For example, consider a 1D representation with 100 items. In our 2D representation, the solutions will be represented as a 10x10 matrix, where each element of the matrix represents an item's location. Again, state-space would amount to 2^n.

Actions: In this formulation, the primary benefit is seen in the action space. From having a representation where the action space increases with the size of the problem, we move to a formulation where the action space is the same for different sized instances. We model the actions as movements in the matrix. We have four (4) discrete actions at any step. Up, Down, Right, and Left. Whenever the agent moves from one cell to another at each step, it reverts an item's selection or deselection decision and moves to another item to make a decision for in the next step.

Reward Function: The reward structure does not change with our structure as well. So, for the 2D formulation, we still use the reward function shown in Equation (5.2).

4.4.6 Main Algorithm

To train and test both 1D and 2D knapsack environments, we use a similar algorithm. The environments differ internally, but the general steps of the algorithm are the same. Combining the above structures, we build our training framework, as shown in Figure 4.4. We use the same testing framework as in the training framework shown in Figure 4.4. Using ten knapsack training instances of different sizes, we process each one in our K-means cluster to get a close-to-optimal feasible solution and also feed these instances to our heuristic to create a 1D ratio representation for each of the instances. Both of these results are used to prepare the DRL environment. Converting the 1D representation of ratios to a 2D matrix and organizing the K-means solution according to the ratios make up our initial DRL environment. We reset the training environment for every new knapsack instance. Each of the MKP instances for training is generated randomly and has different sizes. Independent of the sizes, the learning is controlled by a set number of steps that we define in the algorithm. Each of the instances is used to train the DRL model until the set number of steps is done. When all training is finished, the model is stored to be used for testing.

For testing, a similar flow to the training algorithm is followed, using a different set of instances for testing. Different from training, we perform three tests. For each of the tests, we generate ten instances for each considered size: small, medium, and large. So, in total, 30 instances are tested based on the same loaded model from the training.

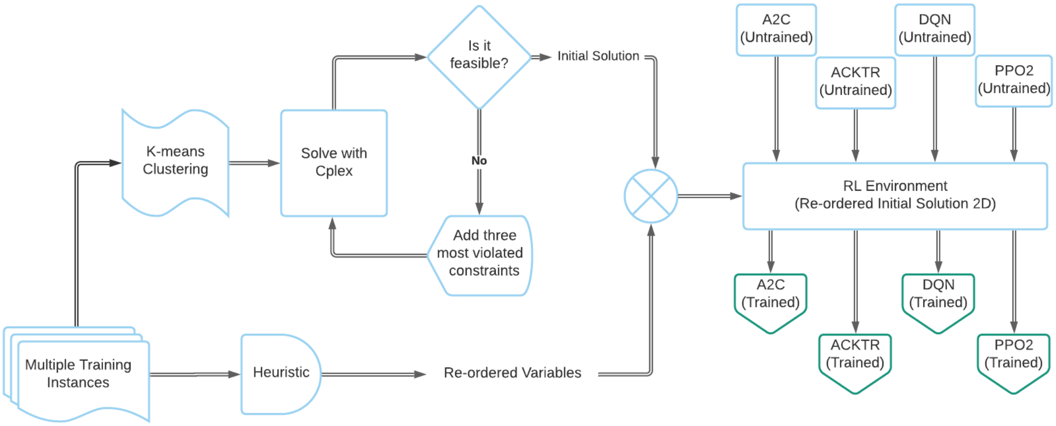


Figure 4.4 Training DRL flowchart.

We demonstrate the steps of the training DRL framework given in Algorithm 3 in Figure 4.4. The steps of the testing DRL framework given in Algorithm 3 are also similar to the training procedure's steps. We use K-means Constraint Clustering (Algorithm 2) to get a close-to-optimal feasible solution and a suboptimal objective value. The variable indices for each instance are re-ordered based on a list of item-worth values, using the heuristic shown in Algorithm 1. Using the initial solution combined with the item worthiness values, we can create the 1D and 2D environments with re-ordered items. After creating the environment, we train our agent for a pre-set number of steps and use one of the training DRL algorithms, A2C, ACKTR, DQN, and PPO2. Internally, 1D and 2D environments have their differences and similarities. When it comes to the state space, reward strategy, and episode concept, they are similar. What changes is the action space. In a 2D environment, we have reduced the number of possible actions in each step to four (4) from n possible actions in a 1D environment, where n is the number of items in the knapsack.

Algorithm 3 describes the training and testing procedures of the DRL framework. Both training and testing procedures make use of Algorithms 1 and 2 to prepare the ground for training DRL agents. The training loop is configured to run a set number of steps for each training instance. Although having the same number of steps, larger instances contribute mostly to the training time. Also, the testing loop is run for a set number of steps for each MKP instance. The number of steps for testing is calculated based on the MKP instance size because of the huge range of the instance sizes we solve. During a set number of steps, multiple episodes can happen. Because of selection and deselection decisions, we keep track of the best solution achieved throughout all episodes.

Algorithm 3 DRL Framework Algorithm

```
**Procedure: Training DRL Framework**
Input: $\mathcal{T}$    {*DRL training steps.*}
Input: $\mathcal{P}$    {*Set of problems P for training.*}
Output: $\mathcal{Q}$    {*DRL trained model.*}
**for** $p \in \mathcal{P}$ **do**
    $\hat{X} \longleftarrow Heuristic$    {*Get 2D ratio representation using heuristic Algorithm 1.*}
    $\bar{X},\ \bar{Z}_{\mathcal{P}} \longleftarrow K$-means    {*Solution and objective obtained using K-means Algorithm 2.*}
    Env $\longleftarrow \bar{X} + \hat{X}$ {*Using heuristic result and K-means initial solution build DRL environment (Env).*}
**end for**
**for each** $\mathcal{T}$ **do**
    Predict action $a \in \mathcal{A}$    {*For each step we only perform one action.*}
    Perform action $a \rightarrow obs, rew, done$  {*After action get the new state (obs), reward (rew), and episode end flag (done).*}
**end for**

**Procedure: Testing DRL Framework**
Input: $\theta$    {*DRL testing steps.*}
Input: $\mathcal{P}_\mathcal{T}, \mathcal{Q}$    {*Set of test problems $P_T$, DRL trained model.*}
Output: $\dot{X}^*,\ \dot{Z}_{\mathcal{P}_\mathcal{T}}$    {*Solution and objective value at the end of the testing procedure.*}
**for** $p \in \mathcal{P}_\mathcal{T}$ **do**
    $\hat{X} \longleftarrow Heuristic$    {*Get 2D ratio representation using Algorithm 1.*}
    $\bar{X},\ \bar{Z}_{\mathcal{P}_\mathcal{T}} \longleftarrow K$-means    {*Solution and objective obtained using Algorithm 2.*}
    Env $\longleftarrow \bar{X} + \hat{X}$    {*Using heuristic result and K-means initial solution build DRL environment.*}
**end for**
**for each** $\theta$ **do**
    Predict action $a \in \mathcal{A}$    {*For each step we only perform one action.*}
    Perform action $a \rightarrow obs, rew, done$  {*After action get the new state (obs), reward (rew), and episode end flag (done).*}
    Store $\dot{X}^*$ and $\dot{Z}_{\mathcal{P}_\mathcal{T}}$    {*Keep the solution and objective if it is the best found.*}
**end for**
$\longleftarrow \dot{X}^*, \dot{Z}_{\mathcal{P}_\mathcal{T}}$    {*Return best solution and objective value from DRL.*}
```

4.4.7 Generalization to Larger Instances

Another essential property for every machine learning model is knowledge transfer. Knowledge/Learning transfer is a machine learning technique where a model trained on one task is re-purposed on a second related task [Goodfellow et al., 2016]. With the current methods, similar instances can be easily implemented, and the agent learns fast in a stable environment. In our model, we do not aim to solve only the same-sized instances. We model our environment in a 30x30 matrix, despite the size of the instances. Our initial environment has all the cells (items) assigned as -1. With this environment, we aim to solve instances as large as 900 items and 900 constraints.

For the large instances, the values of -1 are replaced by assigning item values of 1 or 0 in all cells. For smaller instances, we put the formulation of the square matrix in the top left corner of the environment as 0 and 1s of an initial state obtained from K-means while leaving the other cells at -1 as initially assigned. Figures 4.5, 4.6, and 4.7 show how the environment is filled on different instance sizes. We expect the agent to learn a path leading to the best solution. In the case of 4.5 and 4.6, we expect the agent to learn not to move around cells assigned with -1. This representation is used to generalize the framework to instances with different sizes as it orders items in the matrix and the RL agent learns to focus on a certain search area that is more advantageous in terms of finding the best set of solutions. In addition to the preprocessing of the instances before creating the RL environment, this formulation also reduces our action space, as mentioned in Section 4.4.5.

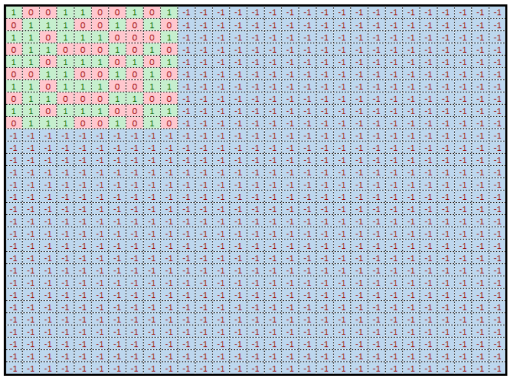


Figure 4.5 Two-dimensional DRL Environment for 100 items and 100 constraints.

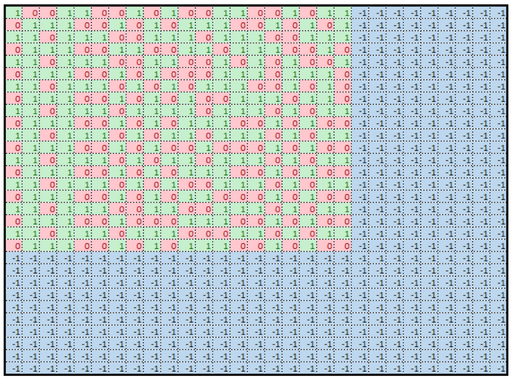
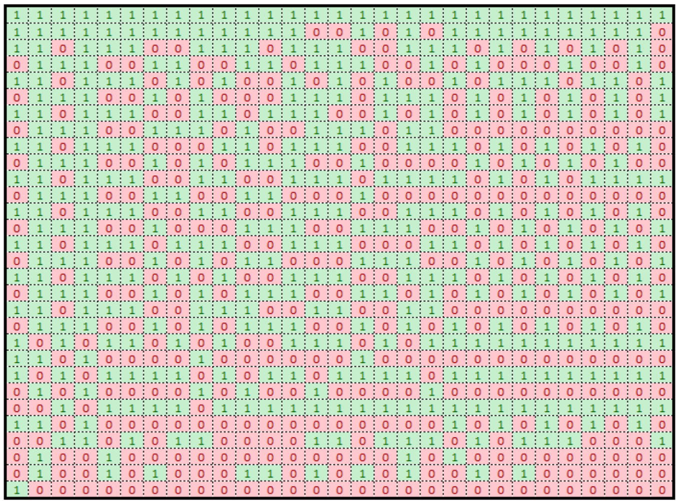


Figure 4.6 Two-dimensional DRL Environment for 400 items and 400 constraints.

4.5 Experiments

4.5.1 Instance Generation and Implementation

To evaluate the computational performance of the DRL algorithm, we generate three types of instances with different sizes. We classify the instances as small, medium, and large based on their sizes. A small instance is composed of 100 items and 100 constraints. A medium instance has 400 items and 400 constraints. Finally, a large instance is made up of 900 items and 900 constraints. For each size of the instance, we



1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	0
1	1	0	1	1	1	0	0	1	1	1	0	1	1	1	0	0	1	1	1	0	1	0	1	0	1	0	1	0
0	1	1	1	0	0	1	1	0	0	1	1	0	1	1	1	0	0	1	0	1	0	0	0	1	0	0	1	0
1	1	0	1	1	1	0	1	0	1	0	0	1	0	1	0	1	0	0	1	0	1	1	1	0	1	1	0	1
0	1	1	1	0	0	1	0	1	0	0	0	1	1	1	0	1	1	1	0	1	0	1	0	1	0	1	0	1
1	1	0	1	1	1	0	0	1	1	0	1	1	1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	1	1	0	0	1	1	1	0	1	0	0	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	1	0	0	0	1	1	0	1	1	1	0	0	1	1	1	0	1	0	1	0	1	0	1	0
0	1	1	1	0	0	1	0	1	0	1	1	1	0	0	1	0	0	0	0	1	0	1	0	1	0	1	0	0
1	1	0	1	1	1	0	0	1	1	0	0	1	1	1	0	1	1	1	1	0	1	0	1	0	1	1	1	1
0	1	1	1	0	0	1	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	1	0	0	1	1	0	0	1	1	1	0	0	1	1	1	0	1	0	1	0	1	0	1	0
0	1	1	1	0	0	1	0	0	0	1	1	1	0	0	1	1	1	0	0	1	0	1	0	1	0	1	0	1
1	1	0	1	1	1	0	1	1	1	0	0	1	1	1	0	0	0	1	1	0	1	0	1	0	1	0	1	0
0	1	1	1	0	0	1	0	1	0	1	1	0	0	0	1	1	1	0	0	1	0	1	0	1	0	1	0	1
1	1	0	1	1	1	0	1	0	1	0	0	1	1	1	0	0	1	1	1	0	1	0	1	0	1	0	1	0
0	1	1	1	0	0	1	0	1	0	1	1	1	0	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1
1	1	0	1	1	1	0	0	1	1	1	0	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	1	0	1	0	1	1	1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
1	0	1	0	1	1	0	1	0	1	0	0	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1
1	1	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	1	1	1	0	1	0	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	1	0	1	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0
0	0	1	1	0	1	0	1	1	0	0	0	0	1	1	0	1	1	1	0	1	0	1	1	1	0	0	0	1
0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	1	0	0	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4.7 Two-dimensional DRL Environment for 900 items and 900 constraints.

generate ten testing instances. The training set of 9 instances is made up of instances of different sizes, with three instances of each size.

We generate each of the test instances for the MKP with the following distributions:

The parameters c_j, a_{ij}, and b_i are independent and identically distributed (i.i.d) random variables sampled from the uniform distribution over $\{1, \ldots, 10\}$, e.g., $U\,[1, R]$, where $R = 10$. We set $b_i = \frac{3}{4}\left(\sum_{j=1}^{n} a_{ij}\right)$.

4.5.2 Implementation Details

We implement our DRL algorithm using Python v3.7 and CPLEX v12.71. Table 4.1 shows the values of parameters used in the heuristic, K-means, and the DRL algorithms with its symbol, its description, and experimental value. We run our K-means algorithm for 30 iterations. We define a different number of clusters in the K-means for each instance size. As the instance size increases, the number of clusters in the K-means algorithm increases as well.

Table 4.1 DRL, K-means, and Heuristic Parameters

Parameter	**Description**	**Value**
n	Number of items	100;400;900
m	Number of constraints	100;400;900
ι	Number of iterations for K-means algorithm	30
Υ	Set of clusters for each instance size	n/25
τ	Number of DRL steps for training	100,000
θ	Number of DRL steps for testing	100
Gap^1	CPLEX optimality gap pre-set for solving the original problem	0.001 %
Gap^2	CPLEX optimality gap pre-set for solving the K-means reduced problem	0.01 %

4.5.3 Results

To report computational results for each considered approach, we describe the following abbreviations:

To report computational results for each considered approach, we describe the following abbreviations:

- **cpx:** solving the original problem (4.1a)-(4.1c) using CLPEX
- **ppo:** training and testing our DRL algorithm using PPO
- **acktr:** training and testing our DRL algorithm using ACKTR
- **a2c:** training and testing our DRL algorithm using A2C
- **dqn:** training and testing our DRL algorithm using DQN
- **obj:** objective function value based on the best solution found for all instances for its specific size averaged over 10 instances
- **ttime:** training time in CPU hours
- **soltime:** solution time in CPU seconds averaged over 10 instances
- **ipred (%):** percentage of item values correctly predicted with respect to the optimal solution averaged over 10 instances

- **gapdiff (%):** the average percentage change in the objective value compared with CPLEX objective $(rl\_obj - cpx\_obj)/cpx\_obj * 100$ where rl_obj and cpx_obj represent the objective found by DRL agent and CPLEX, respectively, averaged over 10 instances. The **gapdiff** value for **cpx** refers to the CPLEX MIP gap.

K-means Algorithm Evaluation To evaluate the performance of our K-means algorithm presented in Section 2, we investigate the rate of improvement and the progress after each CPLEX loop. Starting with clustered constraints, we solve our relaxation, namely K-means reduced problem, and check for violated constraints and calculate the gap between the original and the reduced problem objectives, as described in Algorithm 2. Since we start with a low number of constraints included from the clusters, the first solution found from the relaxed problem is often infeasible for the original problem. Figures 4.8 to 4.13 show how the percent gap between the original and the K-means reduced the problem's objectives, and the number of violated constraints changes after each loop until the point where a subset of constraints is reached to result in feasibility in the original problem. For each instance size, we can see that this solution helps us identify a feasible solution (with a small gap on large instances) in a timely manner.

Figures 4.8, 4.10, and 4.12 show how the gap changes between the objectives of the original and the K-means reduced problems after each iteration for small, medium, and large instances, respectively. For small instances, an optimal solution is reached in most cases, but for medium and large instances, when a feasible solution is found, its objective value has a positive gap of around 0.5% compared to the original problem best objective value found by CPLEX.

Figures 4.9, 4.11, and 4.13 also show how the number of violated constraints changes over each iteration of the K-means algorithm for small, medium, and large instances, respectively. Initially, the K-means algorithm starts with violating a big subset of the constraints but very fast, it converges to a feasible solution. We will use the solution found by K-means to train the DRL agent. The performance of the agent's learning with respect to the initial K-means solution is discussed in the next section.

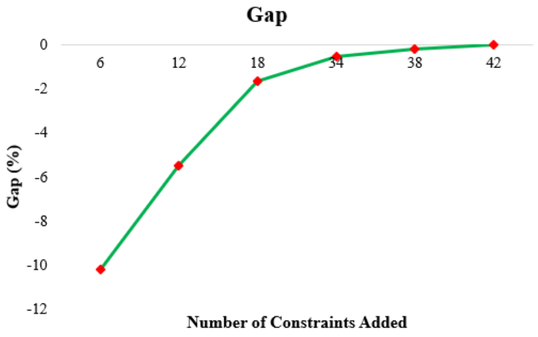


Figure 4.8 Percent gap for small instances.

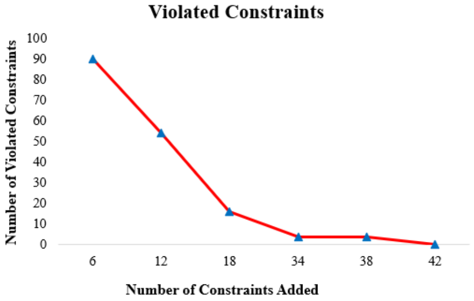


Figure 4.9 Violated constraints for small instances.

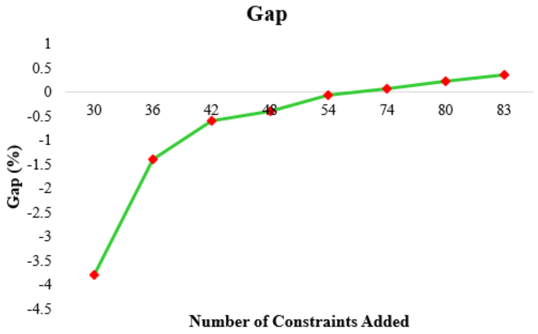


Figure 4.10 Percent gap for medium instances.

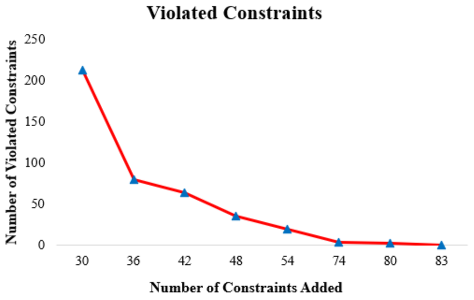


Figure 4.11 Violated constraints for medium instances.

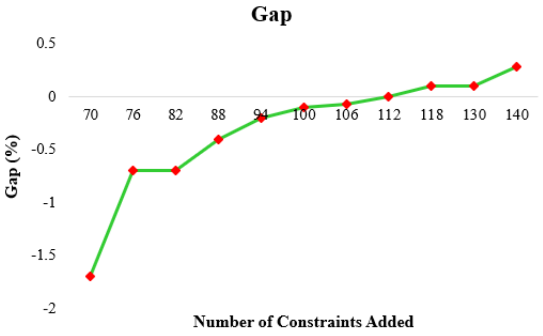


Figure 4.12 Percent gap for large instances.

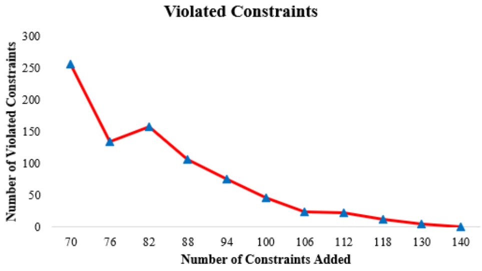


Figure 4.13 Violated constraints for large instances.

Instance Learning and Reward Performance To evaluate the learning process of an agent and the performance of our reward function, we plot reward results of the training process against respective values of the gap between DRL's updated objective value and the objective value of the initial solution provided by the K-means algorithm using CPLEX. We use the initial solution and initial objective value found by the K-means approach given in Algorithm 2 to guide the DRL agents in the learning process. Figure 4.14 shows how the reward function reacts to the percent gap between the DRL solution and the K-means solution. The top y-axis presents the percent gap between the DRL and initial K-means objectives, while the down y-axis shows the reward for each respective episode where a negative reward represents a positive contribution to reducing the objective function. Notice in Figure 4.14 that as the gap increases, the rewards given to the agent decrease. However, the improvement is not completely smooth during the training episodes due to training using different sized instances. We can still see that the model is learning to get higher rewards and, consequently, getting close to or better than the initial objective value found by the K-means algorithm.

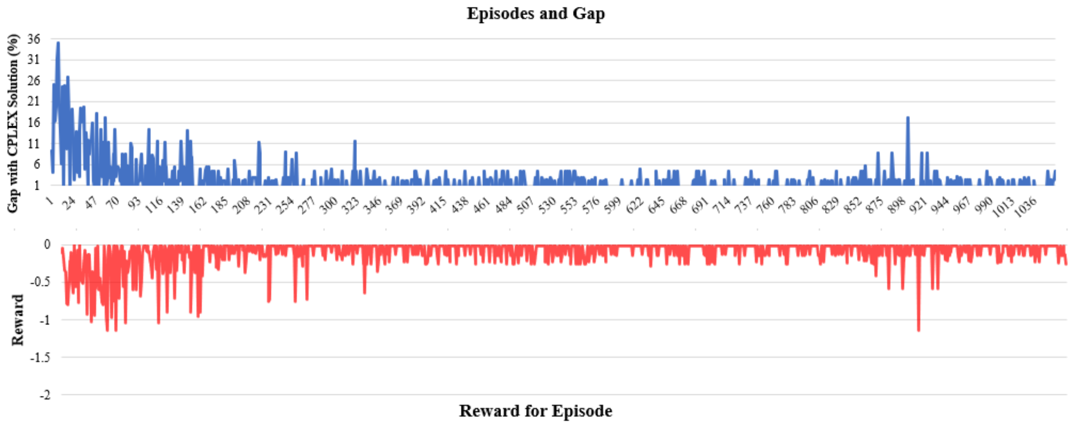


Figure 4.14 Training gap and rewards for each episode where a negative reward represents a positive contribution to reducing the minimization objective function.

As with general applications of the DRL, the learning or training process is the most time-consuming. To evaluate results regarding generalization, we only train one model for each of the DRL algorithms (**a2c**, **acktr**, **dqn**, and **ppo**) where the parameters of the MKP model are sampled using $U[1, R]$, where $R = 10$. Then, each of these DRL models will be used to predict MKP instances of different sizes and distributions. The training time (**ttime**) for **a2c** and **dqn** is 39 CPU hours, while for **ppo** and **acktr** the training takes 38 and 33 CPU hours, respectively.

Comparing Four DRL Algorithms and CPLEX Performances In this section, we compare the performance of each DRL algorithm with that of the CPLEX commercial solver. Each row of Table 4.2 shows results that are averaged over 10 test instances with a specific instance size.

The agent is only trained once for each DLR algorithms (a2c, acktr, dqn, and ppo), and then we predict small, medium, and large instances with the same respective agent. The percentage difference (**gapdiff**) is calculated with respect to the best solution found by CPLEX. For results in Table 4.2, the gap (**gapdiff**) and solution time (**soltime**) depend on the number of iterations used in the K-means algorithm ι and the number of DRL steps used for testing θ.

Our primary aim in this study is to benefit in solving harder instances faster, but we present results for small instances to show a wide range of applicability of our approach. For small instances, the DRL agents are able to find an optimal solution, although each of the agents takes longer to solve than **cpx**. Also, the percentage of item values correctly predicted with respect to the optimal solution is consistently above 97%. For medium instances, we notice that we have small gaps with respect to **cpx**, but they do come with a significant improvement in the solution time. In terms of the solution time and gap with respect to **cpx**, **acktr** is a clear winner for medium instances. Each DRL solution requires at least 45 times less CPU solution time in CPU seconds and only has a max **gapdiff** of 0.28 %. Despite that, **acktr** is closely followed by the other algorithms with a slightly larger gap and solution time. With respect to the number of item values correctly predicted, **a2c** is slightly ahead with a small percentage.

For large instances, the trained DRL agents still show some gaps, but in the worst-case scenario, we improve ten-fold over **cpx** in terms of solution time. The **dqn** seems a clear winner over other DRL algorithms with respect to the solution time, gap, and percentage of item values correctly predicted. The **dqn** provides a solution predicting 97.8% of the same items as **cpx** using at least 14 times less time and only having a **gapdiff** of 0.22%.

In this section, we address the generalization of the proposed DRL algorithms by training each of the agents with a mix of varying sized instances and then using that agent to predict instances with different sizes. In the next section, we show the generalization of our DRL approach to solving instances with not only different sizes but also different distributions.

Generalization to Different Distributions To expand the generalization further, we want to use the same trained agents to predict instances generated from different distributions. For this set of generalization experiments, we generate groups of 10 small, medium, and large instances with a uniform distribution $U[1, R]$, where $R = 25$ and $R = 100$ to test the trained DRL models.

Each row of Table 4.3 shows test results for an average of 10 instances generated with $U[1, R]$, where $R = 25$, using the DRL models that are formerly trained utilizing instances with $U[1 : R]$, where $R = 10$. Thus, for these generalization experiments, we do not perform any additional training. Our results show that for small instances, the DRL framework finds the optimal solution using each of the four RL algorithms, and a high percentage of the items are predicted to be the same as the solution

Table 4.2 Comparison of DRL Algorithms and CPLEX Performances

Instance	**Algorithm**	**obj**	**soltime (CPU sec.)**	**gapdiff %**	**ipred %**
Small	**cpx**	359.9	4	0	-
	a2c	359.9	7	0	98.2
	acktr	359.9	8	0	97.8
	dqn	359.9	5	0	97.0
	ppo2	359.9	7	0	97.2
Medium	**cpx**	1355.1	7206	0.26	-
	a2c	1358.8	159	0.27	97.6
	acktr	1358.7	135	0.26	96.8
	dqn	1398.9	136	0.28	96.6
	ppo2	1358.8	147	0.27	96.8
Large	**cpx**	2994.2	7209	0.25	-
	a2c	3002.9	715	0.29	97.4
	acktr	3002.8	607	0.28	97.3
	dqn	3002.6	494	0.22	97.8
	ppo2	3002.9	680	0.29	97.2

from **cpx**. In medium and large instances, we notice a slight increase in the **cpx** optimality gap and the solution time compared to similar results shown in Table 4.2, implying that those testing instances with $R = 25$ are harder than the training instances with $R = 10$. Despite the change in distribution (R = 10 to R = 25), the gap (**gapdiff**) and the percentage of item values correctly predicted (**ipred**) still perform good. Specifically, the **gapdiff (%)** values provided by DRL algorithms are improved in Table 4.3 compared to Table 4.2. This shows that the solution found by our DRL algorithm gets closer to the CPLEX solution for those harder instances when evaluating the **gapdiff**. Thus, using a distribution for the test instances than those used in training instances does not impact the good solution performance of our DRL approach.

Table 4.3 Comparison of DRL Algorithms with CPLEX for Test Instances with $R = 25$

Instance	Algorithm	obj	soltime (CPU sec.)	gapdiff %	ipred %
Small	**cpx**	818.5	4	0	-
	a2c	818.5	7	0	97.2
	acktr	818.5	8	0	98.6
	dqn	818.5	5	0	98.6
	ppo2	818.5	6	0	98.6
Medium	**cpx**	3157.6	7206	0.43	-
	a2c	3166.5	203	0.28	95.5
	acktr	3164.8	176	0.22	96.0
	dqn	3163.5	177	0.18	96.0
	ppo2	3163.5	173	0.18	96.3
Large	**cpx**	6811.1	7209	0.33	-
	a2c	6826.0	727	0.22	96.9
	acktr	6824.7	637	0.20	97.1
	dqn	6822.7	533	0.17	97.7
	ppo2	6828.5	721	0.25	97.4

Table 4.4 shows further results for our generalization to different distributions. Again, using models trained with instances with $U[1, R]$, where $R = 10$, we predict ten instances for instances generated with $U[1, R]$, where $R = 100$ and varying sizes. Once more, our results show that the DRL framework can scale to different distributions without any significant loss in **soltime** or **gapdiff**. Although we notice an increase

in **soltime** for small instances, the percentage of the item values predicted correctly **ipred** is better compared to those in Tables 4.2 and 4.3. In the case of medium and large instances, despite a small increase in **soltime**, the significant improvement with respect to **cpx** is still maintained. For large instances, while **soltime** increases, at least a 7-fold improvement by DRL algorithms is preserved. Hence, our generalization experiments prove that the DRL framework retains a good performance even if the distribution of the test instances is varied.

Table 4.4 Comparison of DRL Algorithms with CPLEX for Test Instances with $R = 100$

Instance	**Algorithm**	**obj**	**soltime (CPU sec.)**	**gapdiff %**	**ipred %**
	cpx	3179.3	6	0	-
	a2c	3179.3	11	0	100
	acktr	3179.3	12	0	100
Small	**dqn**	3179.3	7	0	100
	ppo2	3179.3	10	0	96.7
	cpx	12209.1	7201	0.36	-
	a2c	12236.6	215	0.22	96.0
	acktr	12699.0	205	0.27	96.0
Medium	**dqn**	12324.9	192	0.24	96.0
	ppo2	12691.6	201	0.21	95.0
	cpx	26864.3	7209	0.35	-
	a2c	26937.3	1012	0.27	97.5
	acktr	26945.0	978	0.30	97.6
Large	**dqn**	26930.3	882	0.24	97.1
	ppo2	26930.6	961	0.24	97.3

RL Partial Prediction To search for better solutions in large instances, we propose a partial prediction that can be used to guide CPLEX and offer a better and more time-efficient solution. By utilizing the DRL framework properties, we can decide on what items the framework selects and deselects with high certainty. Based on this information, we can tailor a partial prediction guided by a threshold level or solely based on the DRL certainty on selection. We show results for three tested partial predictions: *default*, *85%*, and *95%*. The default partial prediction only fixes items that the DRL agents predict with high confidence. This is done by ordering the solution of the DRL agent according to the worthiness of items defined by the Knapsack Transformation Heuristic given in Algorithm 1. Such a solution would have the form $1, 1, 1, 1, 1, \ldots, 0, 1, 1, 0, 0, 1, \ldots, 0, 0, 0$. The *default* prediction is defined by selecting items starting from the left with the most rated until a deselected item is found and starting from the right with the least rated item until a selected item is found. Our goal is to have the DRL agent predict only a subset of all items and let CPLEX work on the uncertainty region of the solution. Once the values of the predicted items are fixed in the optimization model (4.1), we solve it by CPLEX at its default settings. Due to this, for different instance sizes, the prediction percentage changes for the *default* method (see, e.g., in Table 4.5, *70%*, *78%*, and *72%*). To predict at the *85%* and *95%* levels, we still fix a set of items from the right and left of the solution vector and then use CPLEX to solve for the remaining items. For *85%* partial prediction, we use *75%* of the predictions starting from the left of the solution vector and *10%* of the predictions starting from the right of the solution vector, while for the *95%* partial prediction, we fix *80%* of the items starting from the left and *15%* starting from the right leaving only *5%* of the total item values for CPLEX to decide.

Table 4.5 shows results for three partial predictions for each instance size and each algorithm used. **PredPerc** is the percentage of items that are predicted using the DRL solution. For small instances that are solved in a few seconds, **cpx** has a time advantage, sometimes even two-fold. However, all the DRL agents are able to find the optimal solution. Notice that the **soltime** for all partial predictions does not increase compared to the **soltime** reported in Table 4.2, despite solving an additional model where the partial predictions are fixed. This is because the additional time to solve each of the models with partial predictions using **cpx** is quite small, on average 0.01 CPU seconds. For medium instances, in the case of 95% prediction, solution time is insignificant. Therefore, no additional time is needed over the DRL algorithm solution time. In the case of 85% prediction, a small amount of time is required for **cpx** to solve it further, but a big improvement is seen with respect to the **gapdiff**. Even on the default case, which takes the longest time among the partial predictions, an average gap of 0.001% with **cpx** is achieved in considerably lower **soltime**. For large instances, default partial prediction not only achieves an average of 0.0005% **gapdiff** with **cpx** but also provides a slightly better solution in the case of **a2c**. Larger instances take more time to solve compared to medium instances using all methods. Among them, 85% consumes more **cpx** time, while 95% partial prediction model with fixed variables uses on average around 10 CPU seconds. Based on our results in Table 4.5, a manager would not benefit a lot from partial predictions at 95% but would gain at least a two-fold decrease on **gapdiff** if they would decide to use 85% partial prediction. Meanwhile, certain decision-makers that need a close-to-optimal solution and would allow for more time to get a better solution might find the *default* partial prediction most useful.

Partial prediction is a useful feature to serve as a trade-off between reductions of computational time and solution gap. In situations requiring a fast solution of large problems, a partial prediction with a high percentage of fixed items can be used, while in situations demanding close-to-optimal solutions a default partial prediction can be the preferable approach.

Table 4.5 Three Levels of Partial Prediction for Four DRL Algorithms and CPLEX

Instance	Algorithm	PredPerc%	obj	soltime (CPU sec.) *	gapdiff %	ipred%
Small	cpx	-	359.9	4	0	-
	a2c	70%	359.9	7	0	98.0
		85%	359.9	7	0	98.0
		95%	359.9	7	0	98.0
	acktr	70%	359.9	8	0	97.4
		85%	359.9	8	0	97.0
		95%	359.9	8	0	97.0
	dqn	72%	359.9	5	0	99.0
		85%	359.9	5	0	98.0
		95%	359.9	5	0	98.0
	ppo2	70%	359.9	7	0	97.6
		85%	359.9	7	0	97.6
		95%	359.9	7	0	97.6
Medium	cpx	-	1355.1	7206	0.26	-
	a2c	78%	1355.5	2284	0.02	97.0
		85%	1356.4	416	0.09	96.0
		95%	1358.8	159	0.27	97.0
	acktr	81%	1355.9	657	0.05	97.4
		85%	1357.9	145	0.2	96.5
		95%	1358.7	135	0.26	96.8
	dqn	75%	1355.2	2112	0.007	97.2
		85%	1356.4	373	0.11	96.6
		95%	1358.9	136	0.28	97.0
	ppo2	73%	1355.4	1235	0.02	96.7
		85%	1357.2	389	0.15	96.0
		95%	1358.8	147	0.27	96.7
Large	cpx	-	2994.2	7209	0.25	-
	a2c	72%	2994.1	5570	-0.003	98.1
		85%	2995.2	2583	0.03	98.0
		95%	2997.1	716.2	0.09	97.4
	acktr	81%	2994.6	2823	0.01	97.6
		85%	2997.1	1730	0.09	97.3
		95%	3002.8	615	0.28	97.3
	dqn	64%	2994.3	2710	0.003	98.1
		85%	2997	1617	0.09	97.6
		95%	3002.6	504	0.22	97.6
	ppo2	72%	2995.2	2896	0.03	97.5
		85%	2996.5	1803	0.07	97.2
		95%	3002.9	688	0.29	97.6

* *Solution time is calculated by summing up the time needed to find a solution with a DRL algorithm and the model solution with fixed predictions using CPLEX.*

4.6 Discussion and Future Work

We present a DRL framework to solve MKP instances of different sizes and distributions. The framework consists of a heuristic to analyze and generalize MKP properties to estimate item worthiness and an unsupervised clustering algorithm based on K-means to reduce problem size and get an initial feasible solution. Four different state-of-the-art RL algorithms are used to learn patterns in a two-dimensional environment where items can be selected or deselected. Our design is based on a lot of testing while identifying fast and efficient ways to feed the main RL learning process.

Our results show that COPs can highly benefit from the power of deep learning methodologies. Specifically, reformulating hard problems into convenient general deep learning environments allows one to generalize over the solution of a broad class of problems, such as MKP. Based on our experiments, DRL agents can learn and generalize solution strategies for the MKP. Furthermore, commercial solvers can benefit from the computational power of deep learning and form hybrid frameworks that reflect the best side of each methodology.

As a general framework, future work can include improvements anywhere on the framework. For example, another way can be found to gain an initial solution and objective value faster. A more accurate representation of the MKP, rather than using the heuristic, could lead to an improvement in time and accuracy. Future improvements could be introduced to the RL parts. The reward function is a key point in training; therefore, a multi-objective reward function could be designed to look at different aspects of a solution, such as a gap, feasibility, or small violations. A new DRL algorithm designed specifically for the MKP environment can improve sequential decision-making and contribute to faster and more accurate prediction as well. Lastly, the developed MKP environment can be further extended to incorporate stochasticity in sequential decision-making.

CHAPTER 5

A SIMULATION-DEEP REINFORCEMENT LEARNING (SIRL) APPROACH FOR EPIDEMIC OPTIMIZATION AND CONTROL

5.1 Introduction

Coronavirus disease of 2019 (COVID-19) epidemic very quickly paralyzed the world as we know it. After starting as a local epidemic, in a short time, it reached across the world and was declared a pandemic by World Health Organization (WHO). As of June 29, 2021, 33,640,572 individuals have been infected, and 604,115 died from the COVID-19 in the U.S. alone. No government had it easy to come up with the regulations and interventions. Apart from the loss of human lives, COVID-19 also caused an economic recession in the world economy. The global stock market has experienced the worst crash since 1987. Jones et al. [2021] and the International Labor Organization [McKeever, 2020] estimated a loss of 400 million full-time jobs across the world. COVID-19 economic impact is also felt in agriculture [Poudel et al., 2020, Gray and Torshizi, 2021], manufacturing [Tareq et al., 2021, Cai and Luo, 2020], arts and sports [BBC, 2020, Committee, 2020], and tourism [Tounta, 2020]. Even the U.S. was hit hard by the disruption of supply chains [Goel et al., 2021, Nikolopoulos et al., 2021], change of lifestyle [Giuntella et al., 2021], and limited resources [Galanakis et al., 2021]. This disruption was accompanied by a huge job loss [Bell and Blanchflower, 2020, Soucheray, 2020]. With COVID-19 effects the economy, many controversies began. On March 11, 2020, WHO declared COVID-19 a pandemic. Former U.S. President Trump declared a national emergency and accepted that they have ”played it down” not to cause panic in public. On March 16, the Trump government puts in place the first interventions stopping gatherings of more than ten people and canceling nonessential trips for the next 15 days. In fear of an economic failure, this is the closest thing to a nationwide shutdown. What is worse, a tentative reopening provoked a harsher spread of infections, spiking up the debates regarding how the government handled the pandemic [Chaudhry et al., 2020, Ashraf, 2020]

At the end of a dark year, hope arose when the first vaccines were introduced. On December 11, 2020, Food and Drug Administration (FDA) authorized the Pfizer-BioNTech vaccine for emergency use, and just a week later, on December 18, Moderna was authorized as well. The government started vaccination using age and comorbidity-based strategies aiming to protect individuals more prone to critical effects. Another discussion arose whether a strategy where the super-spreaders, individuals contributing the most to the spread of the virus, were targeted first or at least were allowed to get vaccinated could have higher benefits. Moghadas et al. [2021] study a vaccination strategy with a delayed second dose. Their experiments show that a delay of 9 weeks for the second dose could avert at least an additional 17.3% infections and reduce deaths by 0.34 per 10,000 population compared to the four-week interval between the two doses. Gupta and Morain [2021] investigate different prioritization approaches and assess the likeliness of those approaches to reduce morbidity and mortality.

We aim to develop an approach incorporating two components: an evaluation mechanism and a decision-maker. An agent-based simulation is a very suitable approach to mimic epidemic spread and population movements and quantify interventions [Shamil et al., 2021, Li et al., 2021, de Mooij et al., 2021]. Hence, we use an agent-based model as an evaluation of the interventions. Kerr et al. [2020] provide an interesting approach of an agent-based model involving interventions and different contact layers for each individual in the population. Considering high usability and advance of reinforcement learning (RL) (see, e.g., Bushaj and Büyüktahtakın [2021], Delarue et al. [2020], Kong et al. [2018]) and the disability of state-of-the-art mathematical optimization models to deal with large populations, we consider using a DRL agent as governing decision-maker that has the ability to intervene in the simulation and apply available measures. To do this, we propose a simulation reinforcement learning (SiRL) where an agent-based simulation model is integrated inside a reinforcement learning environment.

The structure of the rest of this chapter includes the related work in Section 5.2, followed by agent-based model (ABM) simulation and deep reinforcement learning (DRL) environment details in Sections 5.3 and 5.4, respectively. Further, in Section 5.5, we integrate the ABM and DRL approaches. Finally, we show our experiments and results in Section 5.6.

5.2 Related Work

In many real-world problems, it is difficult to obtain necessary data and reproduce situations. Hence, simulation has always been a very useful methodology to express the environment with its all variables and dynamics. The choice of modeling is highly dependent on the type of the problem, complexity, and the decision makers' requirements. Among simulation methods used in literature are System Dynamics (SD) [Borshchev and Filippov, 2004], Agent-Based Modelling (ABM)[Macal and North, 2009, Epstein, 2009, Macal and North, 2005], Discrete Event Simulation (DES) [Goldsman et al., 2010, Fishman, 2013], and Hybrid Simulation (HS) [Brailsford et al., 2019, Mustafee et al., 2017].

Agent-based simulation has emerged and matured over the last 20 years, expanding both its realm of applications and its sophistication as technology and computing have improved. Agent-based simulation can be utilized to predict epidemic trends and dynamics [Müller et al., 2021, Kieu et al., 2020], evaluate containment strategies and intervention decisions [Kerr et al., 2020, Shamil et al., 2021, Gharakhanlou and Hooshangi, 2020, Hinch et al., 2020, Alzu'bi et al., 2021], and mitigate risks of reopening [D'Orazio et al., 2020, Li et al., 2021].

Agent-based modeling can be very useful to represent real-world interactions of populations and offer a decision-maker the chance to intervene and evaluate the outcomes of each decision. Epstein [2009] suggests that ABM is perfectly suitable for modeling the dynamics of an epidemic across a population. In the context of COVID-19, agent-based modeling has the potential to assist public health officials in responding to outbreaks with an appropriate level of intervention while minimizing the economic impact of those restrictions.

Among recent ABM simulations, Covasim (COVID-19 Agent-based Simulation) [Kerr et al., 2020] models the dynamics of COVID-19 spread in a population by considering demographics based on age, different transmissions among contact layers, and specific viral properties of the disease itself. Covasim is very useful in simulating disease spread and comparing the simulation with other offered non-pharmaceutical interventions such as social distancing, reducing contacts, testing, contact tracing, and quarantining. Li et al. [2021] extend the above study by also implementing a vaccination strategy and performing simulations according to Operation Warp Speed (an intervention proposed by the former Trump administration) to facilitate and accelerate the development, manufacturing, and distribution of vaccines and diagnostics and the plan of one million vaccines per day, proposed by the Biden administration. During the current pandemic, different countries have tried to implement measures to deal with the epidemics having scarce medical resources and aiming to lower the spread and damages, those in human lives and economic aspects as well. Most countries have tried to keep a balance and optimize decision-making based on their available resources.

In addition to simulation studies that approximate the dynamics of an epidemic, mathematical optimization has often been used for decision-making to control epidemic outbreaks. In an epidemic situation, proper resource allocation contributes to a lot of saved lives as well as to a healthier economy. Different mathematical programming methodologies are presented to tackle the resource allocation challenges in a pandemic, such as mixed-integer programming [Büyüktahtakın et al., 2018a], multi-stage stochastic programs [Yin and Büyüktahtakin, 2021a, Bushaj et al., 2021b,a, Yin and Büyüktahtakin, 2021b, Kıbış et al., 2021], and stochastic programs [Coşgun and Büyüktahtakın, 2018, Tanner et al., 2008, Mehrotra et al., 2020]. Dasaklis et al. [2012] critically review the roles of logistics operations and their management on epidemic control and identify possible literature gaps. They claim that the issue of epidemic control in the supply chain literature is fragmented. Most of the available frameworks have very little correlation to the real-case scenarios, and the applicability of the modeling approaches is limited. Queiroz et al. [2020] prepare a detailed review on the impacts of epidemic outbreaks in supply chains and present a series of open research questions to frame a research agenda for scholars and practitioners. In addition, they identify multiple suitable approaches to support supply chain responsiveness, adaptation, and sustainability. Among others, they claim that a combination of simulation theory with dynamic capabilities, could make up for complex scenarios to cope with resource sparsity and sequential decisions throughout the pandemic.

Büyüktahtakın et al. [2018a] propose a mixed-integer programming formulation that integrates epidemic dynamics into a logistics model to project the disease growth while minimizing the total number of infections and fatalities from the Ebola outbreak in West Africa. They provide insights regarding the intervention timing and intensity for each region in Guinea, Liberia, and Sierra Leone. Yin and Büyüktahtakin [2021a] present a multi-stage stochastic programming compartmental model to tackle the uncertain disease progression and resource allocation in an infectious outbreak. They introduce equity constraints in their model and apply them to Ebola disease spread in West Africa. Yin et al. [2021] present a risk-averse multi-stage stochastic epidemics-ventilator-logistics compartmental model addressing the resource allocation changes of COVID-19. Their results show that the short term migration significantly influences disease transmission. Ventilator allocation depends on multiple factors, including initial infections, ICU capacity, the population of a geographic location, and the availability of the ventilators.

Optimization models that oversee the impact of all possible interventions and budget allocation scenarios on the growth of the disease simultaneously (see, e.g., Büyüktahtakın et al. [2018a], Yin and Büyüktahtakin [2021a], Bushaj et al. [2021b,a], Yin and Büyüktahtakin [2021b]) are powerful tools to model epidemic logistics and optimize decision strategies for resource allocation. Such operations research (OR) approaches focus on modeling disease dynamics on a large-scale population over multiple regions and time periods. However, optimization models in combination with agent-based simulations can be extremely difficult to solve. When we focus on a specific population and heterogeneity among disease compartments such as age-specific transmission rates, agent-based models could capture individual-level interactions and detailed disease dynamics better than mathematical programming models. In that case, agent-based models should be supported by a powerful optimization tool.

Deep Reinforcement Learning (DRL) has lately been very attractive to evaluate optimal policies based on a situation. In the last decade, Reinforcement Learning (RL) shifted from the use of tabular formats of actions and states [Watkins and Dayan, 1992, Hasselt, 2010] to the usage of Deep Neural Networks (DNN) due to their immense benefits. The use of DNN in RL has led to advances, such as Deep Q-Learning [Schaul et al., 2015], Double Deep Q-Learning [Van Hasselt et al., 2016], Actor-Critic Methods [Mnih et al., 2016, Wu et al., 2017]. DRL has proven its strength in various applications such as games [Mnih et al., 2015, 2013, Silver et al., 2018], combinatorial optimization [Bushaj and Büyüktahtakın, 2021, Delarue et al., 2020, Li and Hu, 2019], healthcare [Mahmud et al., 2018, Johnson et al., 2016], financial and business management Hu and Lin [2019], Zhang et al. [2017], autonomous driving [Sallab et al., 2017, Chen et al., 2019].

Due to the devastating COVID-19 pandemic, recent studies have already looked up at DRL to help on different applications related to COVID-19 [Kompella et al., 2020, Wan et al., 2020, Bednarski et al., 2020, Awasthi et al., 2020]. Kompella et al. [2020] aim to use RL to optimize decisions during the pandemic in a way that minimizes the economic impact and keeps the hospitals in a normal capacity. Bednarski et al. [2020] investigate the use of deep learning models to provide near-optimal distribution of healthcare equipment to better deal with public response crises similar to COVID-19. Awasthi et al. [2020] tackle the problem of distributing a limited vaccine supply by using a sequential decision strategy based on RL. They propose VacSIM that formulates sequential decision-making into a Contextual Bandits approach to optimize the distribution of the COVID-19 vaccine. They claim that up to 9,039 additional lives could be saved when evaluating their policy against a naive distribution policy. Ohi et al. [2020] implement a DRL agent based on a short-term memory DDQN to learn an optimal policy for maintaining a balance between mitigating epidemic spread and economic cost.

In essence, Simulation Optimization (SO) is the optimization of an objective subject so some constraints while being evaluated using a simulation. Gillisa et al. [2021] propose a simulation-optimization framework that combines an age-based SEIR compartmental simulation model and a genetic algorithm to discover good strategies and optimize intervention strategies. They extract insights from the COVID-19 pandemic to aid policymakers in the closure, protection, and travel decisions by minimizing the total number of infections under a limited budget. Their results highlight that social distancing and wearing masks are of the highest importance, while closures and travel restrictions are more flexible policy restrictions. Onal et al. [2021] present a simulation-optimization framework that searches and treats invasive species under a limited budget. The simulation is responsible for representing the growth of the invader spatially for up to 25 years, and then the optimization model finds an optimal search and path such that it minimizes the economic damage caused by the invader.

Simulation-optimization studies using agent-based modeling can be very effective but often suffer from the dimensionality curse. Specifically, applying those models to a large population not only simulation becomes more challenging, but also the optimization might be impossible. To overcome this challenge, recent studies have used RL-based techniques to utilize the information obtained from the agent-based simulation. Several studies present distinct frameworks combining agent-based models with DRL tools to explore decision-making options. Ohi et al. [2020] implement a simulation model which serves as a virtual environment for training a DRL agent to take non-pharmaceutical decisions based on a specific situation of the epidemic. They demonstrate how agents select possible available actions to reduce the spread of the disease while still considering the economic factors. They present different lockdown strategies that the DRL agent undertakes to halt the resurgence of the disease. Kompella et al. [2020] present a pandemic simulator that models the epidemic spread, including the interactions between individuals in a community, testing with false positive/negative rates, imperfect public adherence to social distancing measures, and contact tracing. They then use an RL-based methodology to optimize mitigation policies within the pandemic simulator.

Inspired by these achievements of DRL, we aim to develop a self-sufficient framework that fully represents the relationship between the evolution of a disease in a population with individual-level interactions and the government's intervention actions to control an outbreak. We propose a Simulation-Deep Reinforcement Learning (SiRL) approach to epidemic disease modeling and decision-making where the simulation is agent-based and optimization is handled by a DRL agent on environment compartmental data. The Covasim methodology of Kerr et al. [2020] has been very successfully used to represent the realism of the COVID-19 pandemic. Hence, we extend the open-source simulation to better fit with the simulation strategy inside our SiRL framework.

5.2.1 Key Contributions

Simulation. In Covasim, all the details for each intervention are defined upfront at the start of the simulation. We extend the Covasim simulation to be flexible towards incorporating interventions in real-time and over multiple time periods. We modify Covasim to incorporate an online intervention at a current time step by feeding the Covasim model with an action from the DRL agent, who represents the decision-maker, at a preset frequency and enforcing the intervention internally based on the details defined. This way, Covasim becomes more flexible, and new interventions can be enforced up to a defined time period. The preset frequency serves as a simulation step size. This step size is set based on a manager's decision-making schedule. If a manager wants to intervene daily, the step size is set as 1, while we can also set the step size to any number.

Another extension to the Covasim model is the incorporation of vaccination strategies. In addition to Covasim's disease progression mechanism, we add vaccination strategies that can be used for any two-shot or single-shot vaccine. Currently, we introduce only vaccines approved under the Emergency Use Authorization (EUA) of the FDA, but an extension to other single-shot or two-shot vaccines can easily be done. An individual can be exposed to other infected individuals at any point during the vaccination process. Depending on the state at which an individual is, we calculate the likeliness to get infected based on the type of the vaccine and how many shots they got. In addition to the age and comorbidity-based vaccination strategy, we also develop a random vaccination strategy where no priorities are set. In the random vaccination strategy, any individual from each group has the same chance of being selected for vaccination, given that they belong to either susceptible or recovered compartments.

Reinforcement Learning. We introduce a Simulation - Reinforcement Learning (SiRL) platform where a governing agent (DRL Agent) evaluates the available information for the epidemic and then takes an action. The action is then applied to the Covasim simulation environment, the state of the population under each health compartment is computed, and the action outcome is quantified as a reward returned to the governing agent. We formulate a multi-objective reward function to provide insights based on a shifted focus of the decision-maker. The multi-objective reward function enables to achieve a trade-off between infections, deaths, and economic stability during an epidemic outbreak.

5.3 Simulation Environment

Throughout the chapter, we refer to some notations and abbreviations, which are defined in Section C.1. To simulate the Covid-19 pandemic in a population of around 9 million people, who reside in New Jersey, we enhance the Covasim model developed by Kerr et al. [2020] (Version 2.1.2, 2021-03-31) and adapt it to our needs and purpose. Kerr et al. [2020] propose an open-source ABM developed to project epidemic trends and explore intervention scenarios. Covasim ABM has many useful features, such as age-structured agents, transmission networks with different social layers such as households, schools, workplaces, and communities. Covasim further includes intrahost viral dynamics with viral-load-based transmissibility. Covasim also supports a wide range of already built interventions such as physical distance, protective equipment, testing, and quarantine, as well as the capability to extend and make custom interventions. Covasim is used extensively in literature for spatio-temporal simulation [Gharakhanlou and Hooshangi, 2020], to derive strategies for non-pharmaceutical interventions [Contreras et al., 2021, Chiu et al., 2020], and to evaluate reopening strategies [Bilinski et al., 2021, Li et al., 2021]

Kerr et al. [2020] also implement a process of calibration calculating the loss using a normalized absolute error. They formulate an equation to find parameters that minimize the function that measures the difference between the observed data and the model predictions. In their calibration module, most of the parameters are fixed based on the values available from literature and only parameter allowed to vary is β, which is the probability of virus transmission when a susceptible individual comes in contact with an infectious individual.

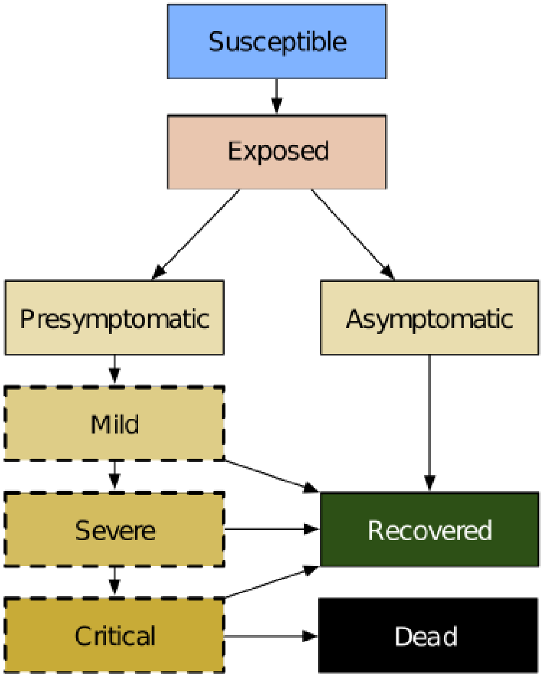


Figure 5.1 Covasim disease progression, compartments, and final outcomes [Kerr et al., 2020].

Figure 5.1 shows the compartmental model structure of Covasim. The yellow shading shows the states at which an individual is infectious and transmits the disease. The susceptible compartment includes all healthy individuals. Once a healthy individual is exposed, they get infected but not yet contagious. As the incubation days are over, an individual either has no symptoms (Asymptomatic) and is recovered, or symptoms start to manifest (Presymptomatic). An individual might experience mild symptoms (Mild) and then transition to the recovered compartment. If the symptoms become severe (Severe), then there is still a chance that the individual will recover, but medical attention might be needed. If symptoms become critical (Critical), then the individual still has a slim chance of recovering, but if not, the individual will be transitioned to the death compartment (Dead). We extend the agent-based simulation in Figure 5.1 to the one in Figure 5.2, where V_1 and V_2 represent the individuals who get the first and second shot vaccination, respectively. In our model, susceptible individuals are eligible to be vaccinated for the first dose. As susceptible individuals wait for available vaccines, they might get exposed to the virus, and depending on the contact, they might get infected. A similar situation is possible for individuals who got the first and second dose of the vaccine. However, the probability of getting infected is way lower due to the protection from the vaccine doses. Some infected individuals might not show symptoms (Asymptomatic), while others (Symptomatic) may show mild symptoms (Mild), but they might also become sicker and need hospitalization (Hospital) or critical (Intensive Care Unit [ICU]). Asymptomatic cases will automatically transition to Recovered after some time. Other symptomatic individuals might worsen and eventually die, but with some probability, those individuals might recover as well. After eight months, antibodies of recovered individuals cannot protect them anymore, so that they will transition to susceptible again [Dan et al., 2021].

In their study, Kerr et al. [2020] portray Covasim as a simulation tool with intervention strategies for the whole simulation predefined at the start of it. And using the multi-simulation feature, they can compare how each intervention affects the disease spread. In our study, we are interested in defining the best intervention strategy for a certain situation of the pandemic periodically over multiple time stages.

Hence, we extend Covasim to be more flexible where an action can be defined at any point in time, and a new intervention will be enforced up to a defined time period. Furthermore, we extend the Covasim interventions by implementing two additional vaccine interventions to perform single-shot or double-shot vaccines. While one of the vaccination strategies considers vaccination with equal probability for each individual, random vaccination model (RVM), the other considers people with comorbidities and older in age with a higher likelihood to get vaccinated than young and healthy individuals, age-based vaccination model (AVM). With this, we aim to provide insights to the discussion regarding the priority to vaccinate for critical individuals or super-spreaders.

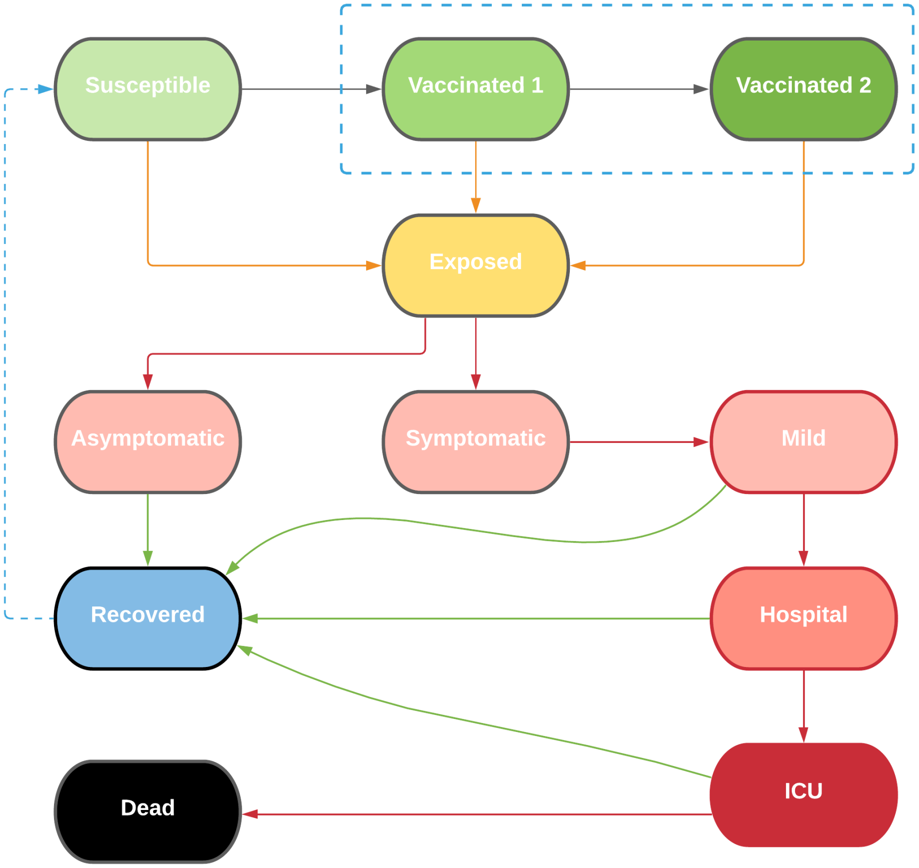


Figure 5.2 Covasim disease progression, compartments, and final outcomes in the extended model.

5.4 DRL Environment

Counting on simulation to describe the compartmental situation of the population, we need to model a DRL environment where the current state of the simulation is represented. We define a state in simulation as the population statistics (percentage of the population on each compartment) in each disease compartment on a particular day during the pandemic. This information is also used to express the state of the RL environment. A state is composed of the following information: the ratio of the susceptible population (S), the ratio of the population who got only the first shot of the vaccine (V_1), the ratio of the population who got both shots of the vaccine (V_2), the ratio of total infections (I), the ratio of hospitalized cases (H), the ratio of individuals in an ICU (C), the ratio of the tested population (T), the ratio of the quarantined population (Q), the ratio of the recovered individuals (R), and the ratio of the dead individuals (D) over all the population.

5.4.1 Episode and States

Episode. We define an episode as the full cycle of simulation and DRL agent intervention decisions-making. Before starting the framework, we define the step size and the full length of the simulation. For example, assuming that we want to simulate for a year and our step size is a month, at the beginning of each month, the DRL agent would enforce interventions in the simulation. Then simulation is run for a month based on the intervention given by the DRL agent. The episode starts with the first intervention of the first month and ends after the simulation for the last month of the year.

States. In our RL environment, we formulate our state as a one-dimensional array containing information for the current compartmental situation of the epidemic and denote it as $\theta := [E_t, S, I, H, C, D, R, V_1, V_2, T, Q]$. A state represents the proportion of the population in each disease compartment defined on Figure 5.2. A state is generated after one simulation run.

5.4.2 Multi-Objective Reward Function

At first, due to the fast spread of the COVID-19 pandemic, many governments were faced with tough choices. COVID-19 started taking lives daily, but most of the governments were slow to enforce closures since they feared the economic collapse Rocha [2020]. In such situations, a government or a decision-maker needs a tool to do a sensitivity analysis and find trade-offs between different objectives, such as reducing the overall disease spread, keeping the economy performing, and protecting people's health, or decreasing the death toll. Particularly, during the COVID-19 pandemic, a full closure would threaten the economy, while no interventions result in more infections, deaths, and a side economic cost related degradation of the quality of human life or loss of lives, workforce reduction, and hospital expenses. A multi-objective analysis could be helpful to evaluate various dimensions of the epidemic's impact simultaneously [Hasan et al., 2019, Cobuloglu and Büyüktahtakın, 2015b, Yin and Büyüktahtakin, 2021a]. We formulate a multi-objective reward function to offer the decision-maker the option of shifting between a strategy to keep the economy flowing to another where they would like to reduce the total death toll. Hence, economic stability and well-being are two main dimensions that make dealing with an epidemic a more difficult challenge. Without considering the epidemic's impact on the economy, which we call the economic index, decision-making would not be complete. Hence, we quantify a contribution for each individual to the economy. Specifically, the health condition of a person defines the level of their contribution to the economy. Quarantining, and work school and business closures come at a high cost. We assume that every healthy individual contributes to the economy a value of 1. In our validation, this contribution is given from susceptible, recovered, and vaccinated individuals. Individuals who get infected will not be able to fully contribute to the economy. Depending on the severity of the infection, it might also become a cost to the economy. Finally, deaths of infected people result in the worst economic loss because the economic contribution of an individual is completely lost.

Economic Index. To quantify the economic situation of a particular day during the pandemic, we formulate Equation (5.1).

$$E_t = S + V_1 + V_2 + R - \alpha \times I - \beta \times H - \gamma \times C - D \tag{5.1}$$

where α, β, and γ can also serve as tuning parameters of the economic index at a time t, and I represents the percentage of infected individuals of all severity including individuals in mild, severe, and critical compartments, D is the percentage of dead individuals, and S, V_1, V_2 are percentages of susceptible, vaccinated with the first shot, and vaccinated with the second shot, respectively.

Multi-Objective Reward Function. Using the formulation of E_t above, we define our multi-objective reward function as:

$$R(\theta) = \lambda \times E_t - \mu \times I - \rho \times D + \pi \times (S + V_1 + V_2) \tag{5.2}$$

where E_t is the economic contribution at time t, and state $\theta := [E_t, S, I, H, C, D, R, V_1, V_2, T, Q]$. Tuning parameters λ, μ, π and ρ are determined based on which part of the objective we want to emphasize more.

5.4.3 Actions or Intervention Measures

Actions. For the learning of our DRL agent, we investigate different possible actions that are realistic but not necessarily exclusive. In practice, we can combine different non-pharmaceutical interventions and vaccines with social distancing measures. In total, we define nine possible actions that our agent can choose from. At the start of the pandemic, we will only include six of these actions as vaccines might not be available at that point. Once the vaccines are available, all nine actions can be applied. The various actions considered are defined below:

0 Do Nothing: We allow the agent to not enforce any restriction on the population.

1 Testing, Contact Tracing, and Quarantine: This action performs tests and traces contacts of positive tests and quarantines them. Usually, these actions go together as the traced contacts are notified and they either get tested too, or they are ordered to remain in quarantine.

2 Close Schools and Non-Essential Workplaces: Governments might decide to close schools and non-essential workplaces and limit gatherings up to a certain number to reduce contacts between the individuals in a population, thus reducing infections and keeping the COVID-19 curve under control.

3 Mandatory Mask: A mandatory mask can be enforced in a population.

4 Testing, Contact Tracing, Quarantine, Close Schools, and Non-Essential Workplaces: Because actions one and two are not exclusive, governments can choose to enforce them at the same time to have a higher impact on slowing the disease spread.

5 Testing, Contact Tracing, Quarantine, and Mandatory Mask: Action one can also be enforced in combination with action three. This action does not close schools or businesses, but it enforces mandatory mask usage to control the spread.

6 Vaccination: When vaccines become available, it is a form of action that can be combined with any non-pharmaceutical measure. This action considers only vaccination, in case governments decide to only use vaccination and reopen without any other enforced intervention.

7 Vaccination and Mandatory Mask: This action consists of a combination of actions three and six. It is seen as a probable reopening strategy as with vaccines the population will become more protected, and masks will reduce transmission of disease.

8 Total Lockdown: In an extreme situation, where the healthcare system has failed, and the government did not intervene timely, a full lockdown might be applied, which enforces all non-pharmaceutical interventions together with vaccination. This measure can result in economic hardships and failures due to the closure of workplaces and businesses.

5.5 Integrated Simulation-RL

Using the ABM simulation and DRL environment presented in Sections 5.3 and 5.4, respectively, we create an Integrated Simulation - RL (SiRL) framework. Figure 5.3 shows how Covasim agent-based simulation interacts with the DRL procedure. We start by creating an RL and an agent-based model environment where compartmental statistics for the population are stored. At the first step, we have an initial information about the compartments. So, the DRL agent takes an action based on the initial proportion of the population in each health compartment. Once the simulation starts, it picks up the decision from the DRL agent, applies the respective intervention, and runs for s days, where s is the step size of the simulation. After s days, the compartmental statistics from the simulation are used to formulate the DRL state and feed it to the DRL agent. Based on the DRL state, we calculate a reward using Equation (5.2). This reward is the evaluation of the last intervention applied by the DRL agent. At this point, we check if the end date of the simulation is reached and based on that the execution of the SiRL framework ends or the DRL agent will take another decision to be applied in the next s days on the simulation environment and follow the same cycle until the end date. When the end date is reached, the episode terminates. Note that the take action in green and dashed yellow box in Figure 5.3 represent the same compartment. The difference is that the action compartment in yellow is executed once in the beginning and then called inside the cycle until the SiRL episode is terminated.

Figure 5.4 describes how the agent-based simulation and the DRL agent interact and exchange information. At time $t = 1$ after we have started the environments (RL and agent-based simulation) and taken the first action, we declare the initial data and start the agent-based simulation incorporating the first action. The simulation will run for $s = 15$ days (our defined step size) and, at the end of the simulation step, will feed compartmental statistics and the economic index to the DRL agent. Based

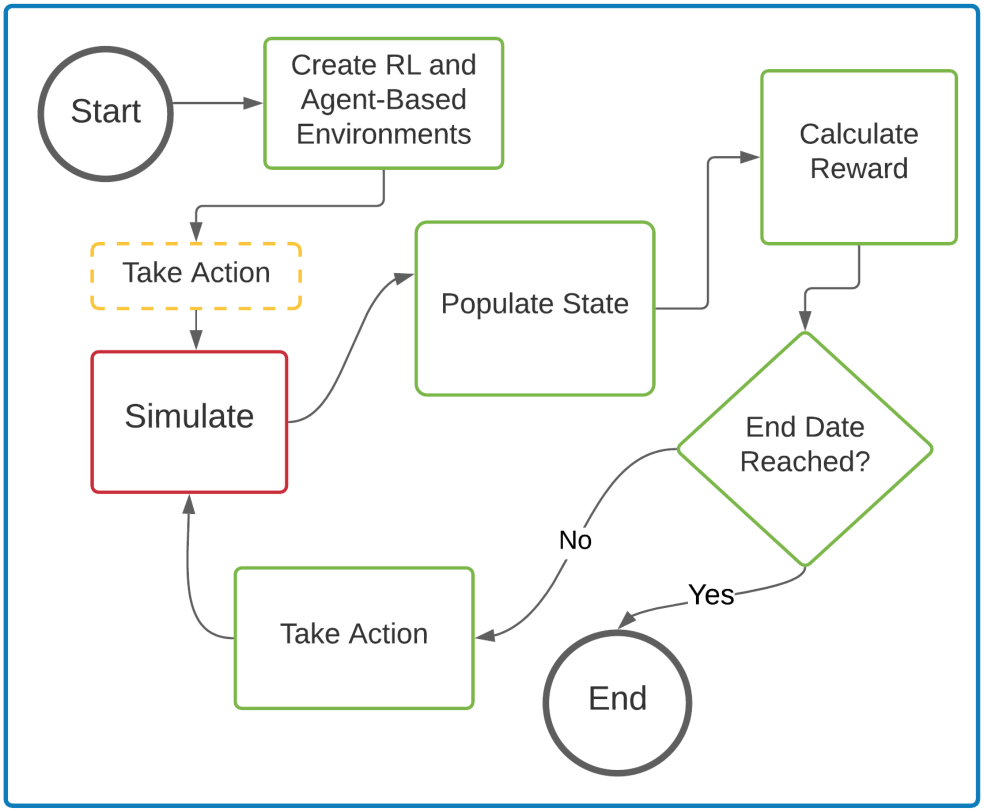


Figure 5.3 The SiRL Framework, which is an agent-based simulation integrated at the heart of a DRL framework.

on the state, the DRL agent takes decision x_1 which enforces interventions on the simulation for the next simulation period. In turn, after this simulation period ends, it will again give the new compartmental statistics after the intervention where the agent's action is evaluated according to the reward function. This process continues until the entire simulation ends.

5.5.1 Training Algorithm

Algorithm 4 describes the general steps and data used to train an agent. First, we create the respective simulation and RL environments. At the start of a simulation, we decide on the total population, the total length of the simulation, and the step size at which we enforce interventions. At the start of the RL environment, we initialize

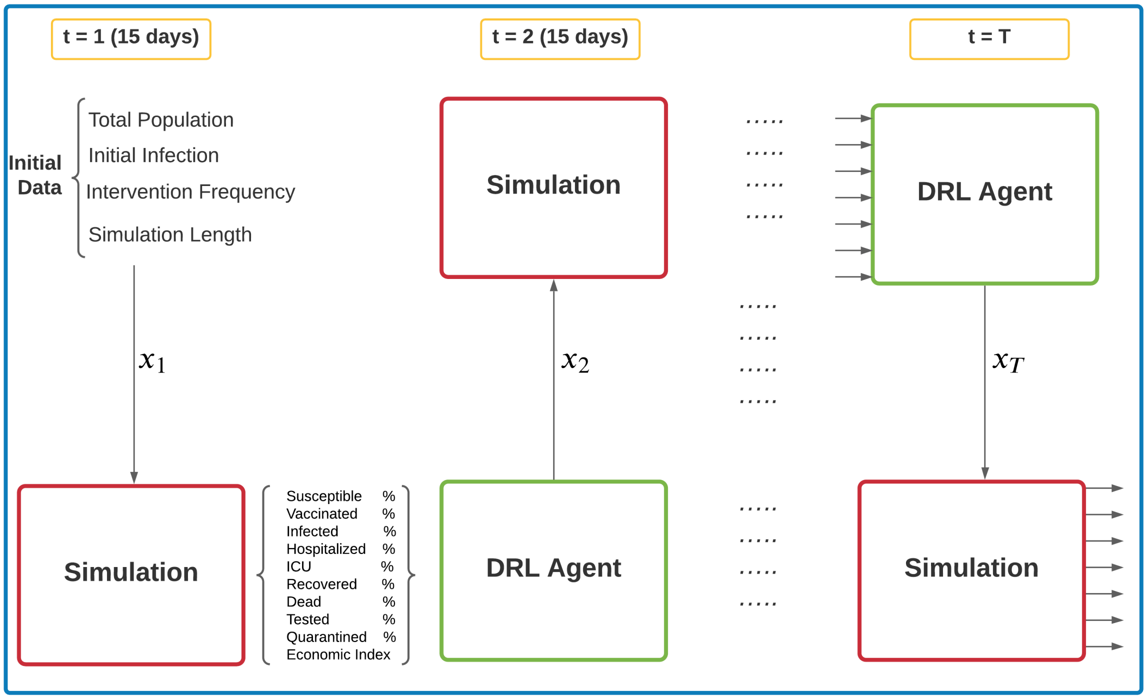


Figure 5.4 Agent-based simulation and DRL agent information exchange between simulation periods.

the agent and define the weights of the reward function. Then for each simulation period, we extract the compartmental statistics from the simulation and feed them to the DRL agent. Compartmental statistics include the percentage of the population in each compartment and the economic index at the end of the simulation period. Having this information, the DRL agent will decide on an action $x_j \in \mathcal{X}$ at simulation run j. Based on this action, the simulation is run for one-step size, and then a reward is generated to quantify how good the action of the agent was.

Algorithm 4 Simulation DRL Training Algorithm

```
**Procedure: SiRL**
Input: $\Delta, s, \sigma, \Theta^0, \theta^0$    {*We start with the total population, step size, simulation periods, initial compartmental statistics, and initial state.*}
Output: $\Omega$    {*Trained Deep Q-Network (DQN) Model.*}
Initiate SiRL with $\Delta, s, \sigma$    {*Create DRL and simulation environments.*}
Take an initial action $x_0$    {*First interventions.*}
**for** $j \in \mathcal{J}$ **do** {for each simulation period}
    $\theta^j \longleftarrow \Theta^j$    {*Update state with compartmental statistics*}
    Take action $x_j \in \mathcal{X}$
    $R(\theta^j, x_j) \longleftarrow$    {*Calculate reward.*}
**end for**
```

5.6 Experiments

In our experiments, we want to be flexible and generalize over different possible epidemics. Therefore, we test different models: one without a vaccine available (no-vaccination model as **NVM**), another after the vaccine is discovered and an age-based vaccination is applied (age-based vaccination **AVM**), and another after the vaccine is discovered but everyone is eligible to be vaccinated above the age of 12 (random vaccination **RVM**). We gather data for the COVID-19 epidemic in New Jersey and address the management of the disease.

5.6.1 Data Gathering

We collect bi-weekly compartmental data from the start of the Covid-19 epidemic until the beginning of our project (March 1, 2020, to April 15, 2021). The compartments we consider are Susceptible (S), Infected (I), Hospitalized (H), ICU (C), Dead (D), Recovered (R), Tested (T), Vaccinated with 1st shot (V_1), and Vaccinated with the 2nd shot (V_2) obtained from CDC database and crosschecked with the NJ COVID-19 dashboard [NJ, 2021]. In addition, to compare and understand decision-making at any point of the pandemic, we also collect government decisions to identify what interventions are active and a specific date. We collect these data for the whole U.S. and the state of NJ in particular.

Figure 5.5 presents a decision timeline for the U.S. during the beginning period of the COVID-19 (March 1, 2020, to June 30, 2020). This timeline also corresponds in close dates with the responses that each state has taken to control the spread.

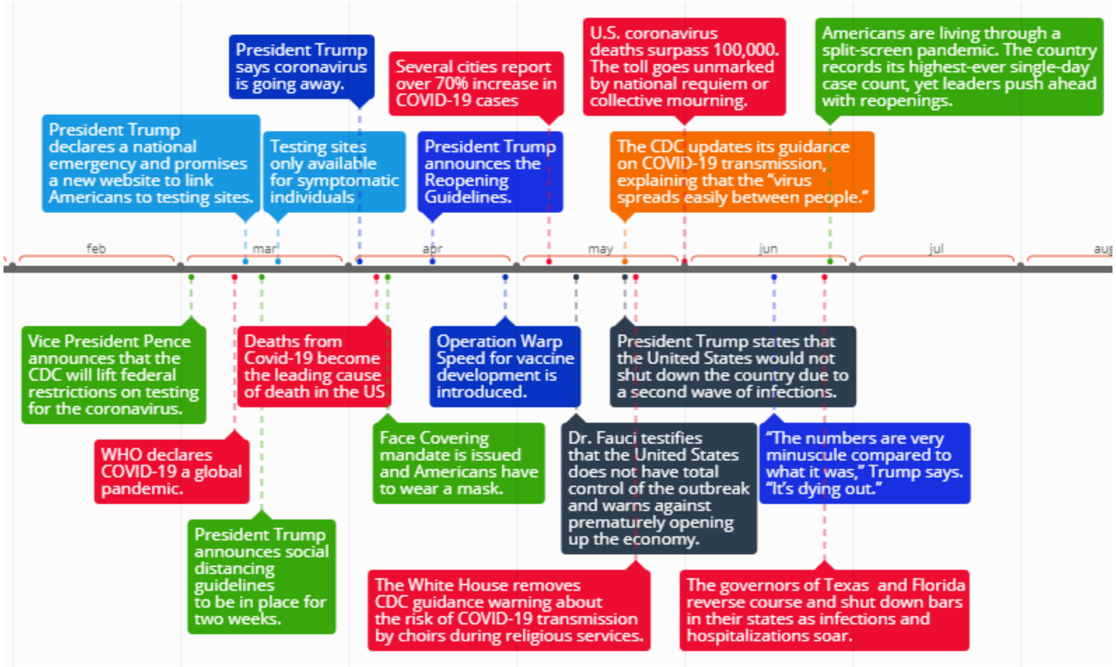

President Trump declares a national emergency and promises a new website to link Americans to testing sites.
President Trump says coronavirus is going away.
Testing sites only available for symptomatic individuals
Several cities report over 70% increase in COVID-19 cases
President Trump announces the Reopening Guidelines.
U.S. coronavirus deaths surpass 100,000. The toll goes unmarked by national requiem or collective mourning.
The CDC updates its guidance on COVID-19 transmission, explaining that the "virus spreads easily between people."
Americans are living through a split-screen pandemic. The country records its highest-ever single-day case count, yet leaders push ahead with reopenings.
feb
mar
apr
may
jun
jul
aug
Vice President Pence announces that the CDC will lift federal restrictions on testing for the coronavirus.
WHO declares COVID-19 a global pandemic.
Deaths from Covid-19 become the leading cause of death in the US
President Trump announces social distancing guidelines to be in place for two weeks.
Face Covering mandate is issued and Americans have to wear a mask.
Operation Warp Speed for vaccine development is introduced.
The White House removes CDC guidance warning about the risk of COVID-19 transmission by choirs during religious services.
Dr. Fauci testifies that the United States does not have total control of the outbreak and warns against prematurely opening up the economy.
President Trump states that the United States would not shut down the country due to a second wave of infections.
"The numbers are very minuscule compared to what it was," Trump says. "It's dying out."
The governors of Texas and Florida reverse course and shut down bars in their states as infections and hospitalizations soar.

Figure 5.5 COVID-19 timeline from April 1, 2020, to June 30, 2020, created based on the information provided in Thebault et al. [2021].

To provide a robust framework, SiRL can be used to extract control measures for different epidemics. In our case, we want to draw conclusions at any point during the COVID-19 pandemic. That is why we calibrate our model and train our DRL agent in different stages of the pandemic. We consider the start of the COVID-19 pandemic where a vaccine is not available, and we refer to this model as the no-vaccination model, **NVM**. We also investigate the COVID-19 dynamics after the vaccines are introduced. With the vaccination models, to be consistent with the reality, we calibrate our model using age and comorbidity-based vaccination strategy, **AVM**. We then use these calibrated actions to implement also a random vaccination strategy, **RVM**.

Intervention Effect Calibration for NVM We use our simulation to measure the effect of non-pharmaceutic risk measures on the COVID-19 progress. Thus, we can use these quantified effects to train our agents. To calibrate between Covasim and New Jersey environments we calibrate each no-vaccination model action by reproducing the COVID-19 spread during its first four months where vaccines were not available. In these four months, we mimic governmental actions in the same period that they were enforced.

During the first four months (March 1 to June 30), the government suffered from resources and not many effective interventions were implemented. On March 3, Vice President Pence announced that CDC would lift federal restrictions on testing for COVID-19. Despite that, until April 12, 2020, it was not easy to get tested. On March 11, 2020, WHO declared COVID-19 a global pandemic. Two days later, on March 13, 2020, President Trump declared a national emergency and promised to increase efforts to make testing available and accessible for Americans. On March 16, 2020, the Trump government also announced social distancing guidelines to be in place for two weeks initially.

Around the same time, on March 18, 2020, Governor Murphy of New Jersey, in an attempt to slow down the spread of the disease, ordered the closure of all pre-K, K-12, higher education institutions, casinos, theaters, gyms, and non-essential retail, recreational and entertainment businesses also banning gatherings of people more than 50.

Figure 5.6 shows the validation of the **NVM** model where we exclude vaccination as an intervention. We present the absolute value of the difference for each compartment between our simulation and the CDC data. The y-axis represents the absolute value of the difference in percentage between the value of each compartment of CDC data and the respective value of the compartment in the simulation. Notice that we are 0.01% away from the real data on a four-month simulation based on the $|T - T_s|$ metric, which refers to the absolute difference between the real treated proportion of the population (T) and simulated treated proportion of the population (T_s) in the worst case. In the best case, the percent difference between the final compartmental statistics of the SiRL at the end of each simulation period and the real data based on the $|C - C_s|$ metric is 0.001%.

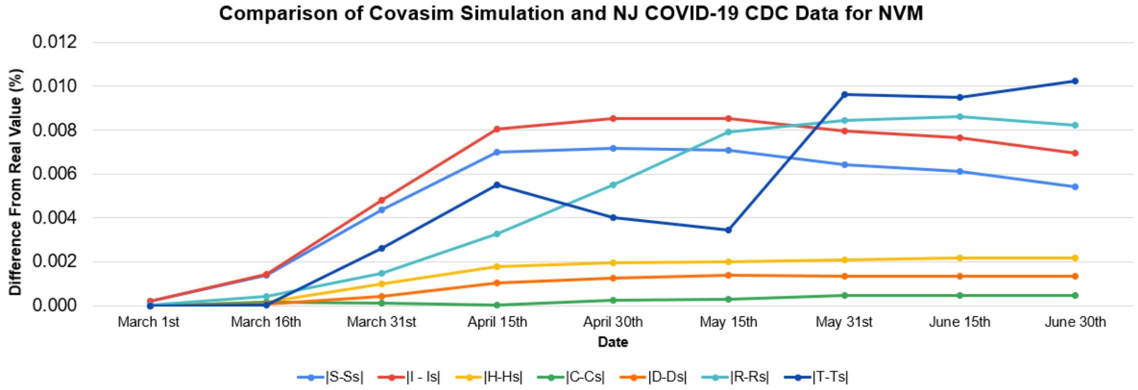


Figure 5.6 Comparison of the Covasim agent-based simulation data and CDC data for the state of New Jersey for the period from March 1, 2020, to June 30, 2020. The x-axis shows the timeline where each dot on the trend lines corresponds to a decision. The y-axis shows the absolute value of the difference between the real number of individuals in each compartment (S, I, H, C, D, R, T) at a point in time and the number of individuals estimated from the simulation in that compartment $(S_s, I_s, H_s, C_s, D_s, R_s, T_s)$. For example, trend line $|S - S_s|$ represents the absolute difference between the real susceptible proportion of the population (S) and simulated susceptible proportion (S_s) at bi-weekly dates starting March 1 to June 30. Similarly, the trend lines for each compartment are plotted.

In addition, we apply the paired t-test to investigate the difference between the mean of the bi-weekly compartmental values of the simulation and the mean actual values provided by the CDC. According to the statistical analysis shown in Table 5.1, our validation is statistically similar to the actual data reported by the CDC.

Intervention Effect Calibration for AVM To model vaccination intervention in our model, we study the period when vaccines became available. On December 11, 2020, The Food and Drug Administration authorized the Pfizer-BioNTech vaccine for emergency use. A week later, on December 18, the Moderna vaccine was also

Table 5.1 Paired T-Test Analysis Comparing the Bi-Weekly Compartmental Data from the NVM Simulation with the Actual Data from the CDC

Compartment	Mean		Two-tailed paired t-test		
	Actual	**Predicted**	**t-stat**	**t-critical**	**p-value**
Infected	107,650	107,569	0.72	2.2	0.49
Hospitalized	22,523	20,890	0.76	2.1	0.46
ICU	966	964	0.73	2.2	0.48
Dead	9,251	9,242	0.67	2.3	0.51

authorized with the same status. Despite this, for the rest of the year 2020, the vaccination campaign is off to a chaotic, confused, and slower-than-expected start, ending up with less than the planned 20 million doses. We model our vaccination models based on the vaccine availability data provided by CDC for Pfizer-BioNTech, Moderna, and Johnson & Johnson vaccines and their respective protection levels by does as research shows. Figure 5.7 shows the validation for the age-based vaccination strategy (**AVM**) compared to the real compartmental data for NJ. Each line represents the absolute value of the difference between the agent-based simulation model and the real NJ CDC Data reported from CDC, including vaccination. We simulate for four months starting from December 15, 2020, to April 15, 2021. During this period, vaccinations were done according to age and comorbidity-based priority. Our validation is off only 0.12 % during the whole four-month simulation period. The actions calibrated using the agent-based simulation for the age-based vaccination strategy (**AVM**) will also be used to run a simulation regarding the random vaccination strategy **RVM**.

Further, we apply the paired t-test to investigate the difference between the mean of the bi-weekly compartmental values of the simulation, including one-shot and two-shot vaccinated individuals, and the mean actual values provided by the CDC. According to the statistical analysis shown in Table 5.2, our validation is statistically similar to the actual data reported by the CDC.

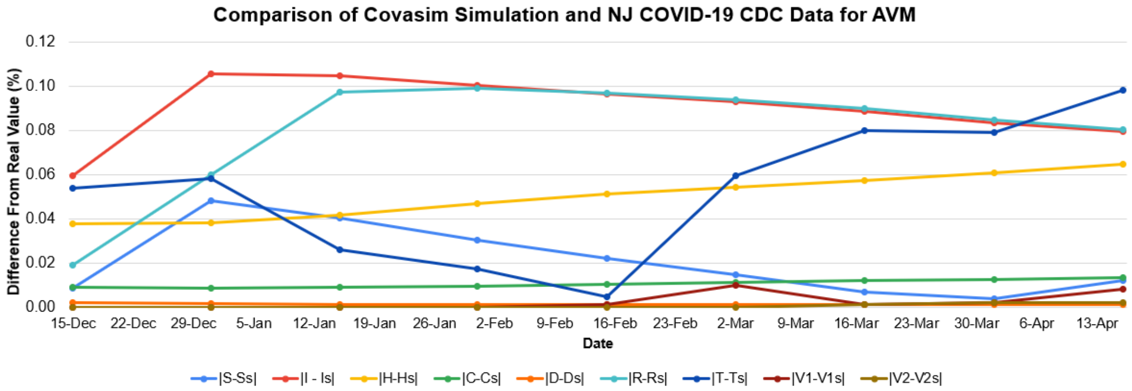


Figure 5.7 Comparison of the Covasim agent-based simulation data and CDC date for the state of New Jersey for the period from December 15, 2020, to April 15, 2021. The x-axis shows the timeline where each dot on the trend lines corresponds to a decision. The y-axis shows the absolute value of the difference between the real number of individuals in a compartment at a point in time and the number of individuals estimated in that compartment from the simulation. For example, trend line $|H - H_s|$ represents the absolute value of the difference between the hospitalized individuals in New Jersey and the hospitalized individuals estimated by the simulation at bi-weekly dates starting December 15, to April 15. Similarly, the trend lines for each compartment are plotted.

The paired t-test for the simulated data using the AVM strategy proves further

that the

Table 5.2 Paired T-Test Analysis Comparing the Bi-Weekly Compartmental Data from the AVM Simulation with the Actual Data from the CDC

Compartment	Mean		Two-tailed paired t-test		
	Actual	Predicted	t-stat	t-critical	p-value
Infected	677,437	6667,457	0.03	2.3	0.97
Hospitalized	556,169	556,452	0.36	2.3	0.72
ICU	124,795	125,920	1.01	2.3	0.40
Dead	21,525	21,510	0.27	2.3	0.79
Vaccinated 1	1,373,739	1,368,578	1.04	2.3	0.35
Vaccinated 2	631,651	630,619	0.31	2.3	0.75

5.6.2 Results

We show results for different periods of the pandemic in the state of New Jersey. We emphasize the usability and flexibility of the framework by a comparative study of different strategies for decision-making during the pandemic.

No-Vaccination Model Training.

Figure 5.8 shows the reward agent gets during training. We train by simulating around 30k episodes, where each is a four-month simulation (March 1 to June 30, 2020). The approximate training time was 30.2 hours. As the agent goes through more episodes, we can notice that it is building a behavior. The learning trend in Figure 5.8 is calculated using a moving average of 15 periods.

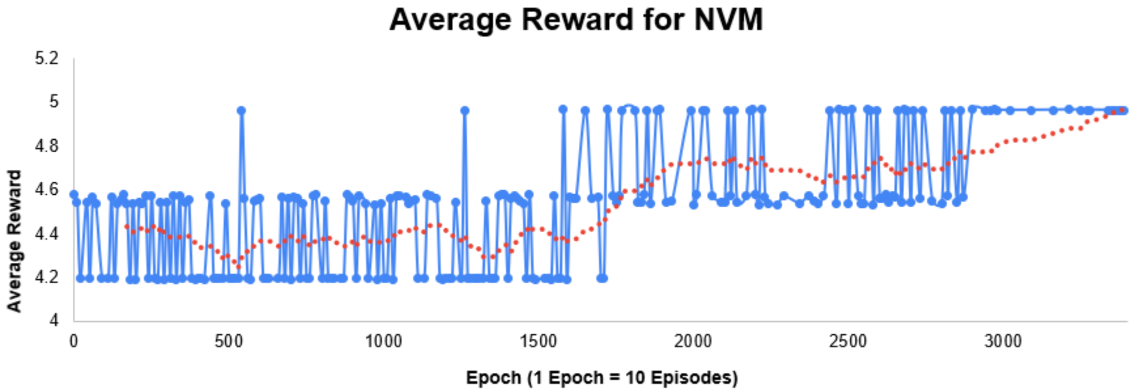


Figure 5.8 Reward for each training epoch for the **NVM** model.

Age-Based Vaccination Model.

Figure 5.9 shows the progress of the training agent for around 30k episodes. An episode is done once a four-month simulation is run (December 15, 2020, to April 15, 2021). The approximate consumed time to train for the **AVM** is 32.7 hours. The trend calculated using a moving average shows an increase as the agent trains in more episodes. This signifies that the agent is learning to win higher rewards, therefore, building knowledge to what action is good in a certain state.

Comparison to Government Actions

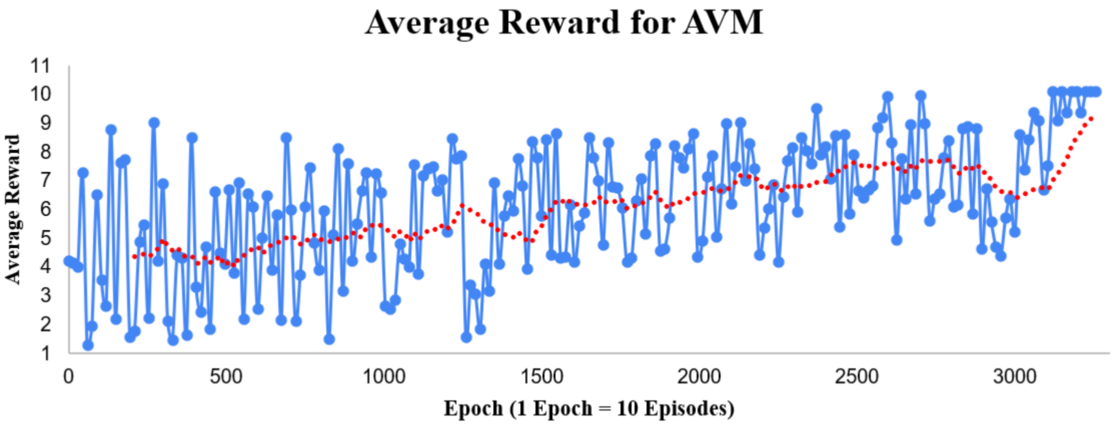


Figure 5.9 Reward for each training epoch for the **AVM** model.

In Section 5.6.1, we calibrate the government decisions during the first four months of the pandemic. After we train, we allocate our DRL agent the same resources and actions to observe what the agent deems optimal. Based on the agent's response, the government closes schools and non-essential workplaces and uses tests to identify infected individuals and then trace their contacts and quarantine them for the first 45 days from March 1 to April 15, 2020. After that is done in the first months, our agent suggests a reopening but enforcing mandatory masks. It is even more interesting that this strategy is very similar to what the government did, but there is a shift in time. Our model suggests contact tracing and stay-at-home orders must have been enforced exactly at the beginning of March and then start reopening around mid-April. This result implies that the government was late in any of the actions except the reopening date. Since measures were not executed timely and sufficiently to control the outbreak, reopening backfired in more cases after the April reopening, which still kept most education institutions closed for the rest of the year. Figure 5.10 compares the decisions taken by the government with the decisions suggested by the trained DRL agent for the NVM model. Above the x-axis, we map the government decision. Notice that there is a delay in action. Testing is announced that it will be available in the first week of March but it was made widely available for symptomatic people around mid-April. Below the x-axis, we describe the suggested actions from the DRL agent. Notice that during the first months testing and contact tracing is important, while also schools and workplaces are temporarily closed to slow the spread down. After that, a reopening is suggested by only enforcing masks.

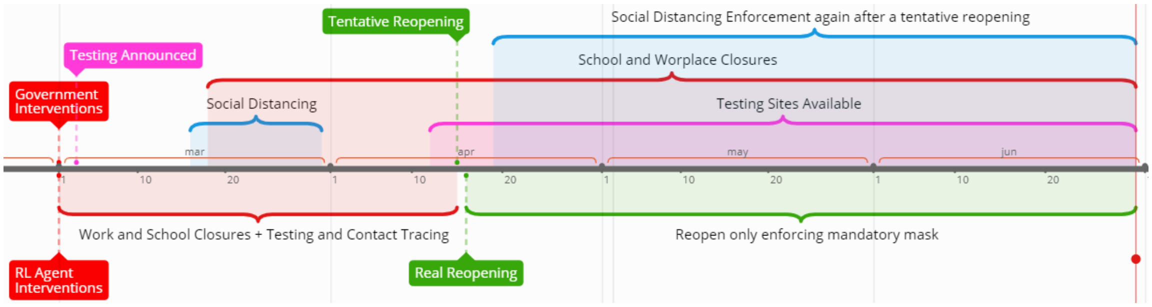


Figure 5.10 Comparison of government actions and DRL agent actions during the first four months of the COVID-19 pandemic in the U.S. Above the x-axis we describe the government actions and below the x-axis the DRL agent suggestions are shown.

Figure 5.11 illustrates the comparison between the NJ and Federal government interventions enforced during the first four months after vaccines were introduced, specifically from December 15, 2020, to April 15, 2021, with the interventions suggested from the DRL agent trained using the age-based vaccination model. Above x-axis notice that the government implemented all available interventions in combination with vaccination. During this period, the government gave priority to older people and those with preexisting conditions. Month after month, the age bar for vaccination was reduced, allowing more younger people to get vaccinated. On April 19, 2021, all individuals older than 16 became eligible for vaccination in New Jersey. In our age-based vaccination strategy, we follow a similar pattern. We give a higher probability to the older ages while reducing it after each month. Our model suggests that testing, contact tracing, and mandatory mask be enforced for the second half of December 2020 and the first half of January 2021. It is interesting that the model does not suggest an immediate vaccination. This happens at the beginning when the vaccine supply was small. So, the DRL agent does not see it as highly beneficial since the number of vaccine doses available was very low when vaccines were first offered. When equal weights are assigned to each sub-objective in the reward function in Equation (5.2), the DRL agent cannot capture that even a very small number of vaccines should be used. From the second half of January 2021, the agent suggests the enforcement of all measures while vaccination should also be applied with those interventions. After only a month, in mid-February, the DRL agent suggests lifting the closures but recommends a continued vaccination while also enforcing the use of masks.

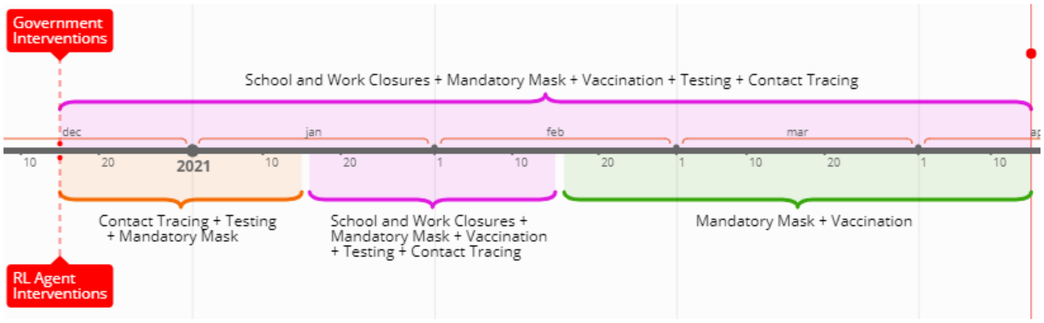


Figure 5.11 Comparison of government and DRL agent actions for the first 4 months after vaccine was introduced for COVID-19 using an age group vaccination strategy. Above the x-axis, we describe the government actions, and below x-axis, the DRL agent suggestions are shown.

Figure 5.12 compares decisions suggested by the DRL agent trained with random vaccination strategy towards those enforced by the government. Differently also from the **AVM** strategy, **RVM** suggests only 15 days of mandatory mask, testing, and contact tracing followed by full closures with vaccination for the next month up until January 31. A reopening is suggested at the beginning of February, combining the mandatory mask with vaccination, which is earlier than that suggested by the **AVM**.

Economic Standing To compare the economic situation in different simulations we use the formula in Equation (5.1). We calculate the economic index by assigning a weight to each of the compartments. Figure 5.13 compares the economic situation between the simulation with the no-vaccination model (**NVM**) and the CDC data for

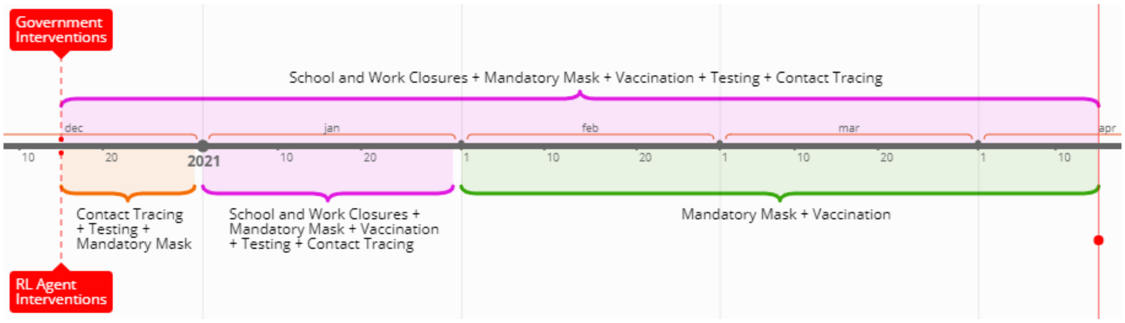

Government Interventions
School and Work Closures + Mandatory Mask + Vaccination + Testing + Contact Tracing
dec
jan
feb
mar
apr
10
20
2021
10
20
1
10
20
1
10
20
1
10
Contact Tracing + Testing + Mandatory Mask
School and Work Closures + Mandatory Mask + Vaccination + Testing + Contact Tracing
Mandatory Mask + Vaccination
RL Agent Interventions

Figure 5.12 COVID-19 timeline allowing all age groups access to vaccination. Above the x-axis, we describe the government actions and below x-axis, the DRL agent suggestions are shown.

the state of New Jersey. Notice that NVM simulation allows for a weaker economic situation since we use equal weights for each sub-objective in the reward function in Equation (5.2). Based on our comparison, our model did decide on reopening, but it is doing slightly worse from the economic point of view. This is due to the multi-objective function as we give equal weight to each of the sub-objectives. If a manager wants to focus more on a flowing economy, a larger weight can be given to the economic sub-objective of the reward function in Equation (5.2).

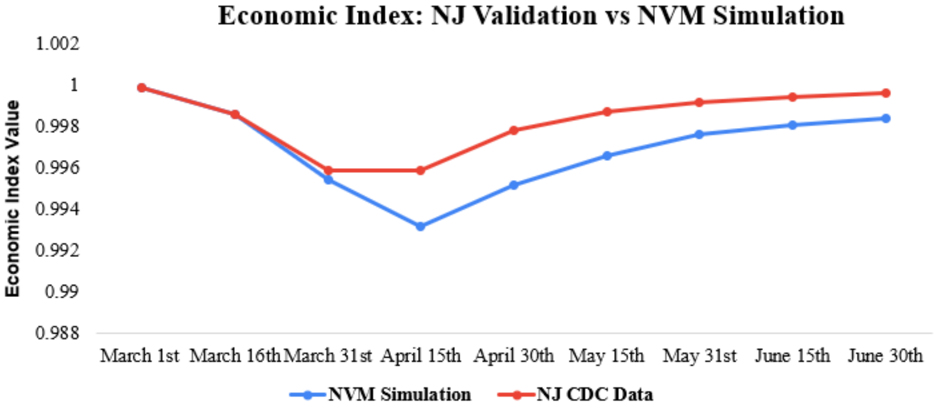


Figure 5.13 Comparison of the economic index between NJ CDC Data and simulation data using NVM.

We also compare the economic situation between the CDC-reported data and our vaccination strategies based on the SiRL framework. Figure 5.14 compares the economic standing at 15-day intervals. We notice that the age-based vaccination maintains slightly the same economy as the CDC data. That is because our results for using an age-based vaccination strategy provide a very good estimation of the real situation. Surprisingly, a better economic standing would be achieved using a model with available vaccination for all individuals older than 12 years old, corresponding to the random vaccination strategy.

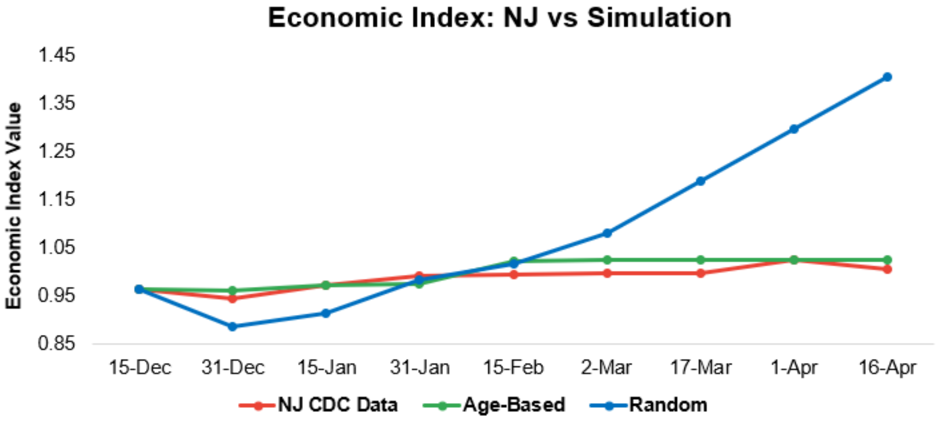


Figure 5.14 Comparison of the economic index between NJ CDC Data and simulation data using AVM and RVM.

Multi-Objective Analysis In this section, we consider modifying the reward function shown in Equation (5.2). Tuning this function will shift importance and suggest decisions to reduce the worst outcomes of different situations. For example, if a government wants to prevent the infection rate and severity of the epidemic, it can give more weight to the μ parameter. To analyze how each tuning parameter affects the compartmental statistics, we consider four formulations with respect to the reward function: economy, death toll, total infections, and healthy individual. For each of these models, we modify the reward function by increasing the respective tuning parameter five times and retrain the model to perform tests. For example, in our experiments, we give a value of one to each of the tuning parameters. When we want to emphasize the economy, we use $\lambda = 5$, while other tuning parameters are still one.

Table 5.3 compares how the number of individuals in each compartment changes with a different focus on the reward multi-objective function. Parameters below the headers in Table 5.3 show the respective values for each tuning parameter of the multi-objective reward function shown in Equation (5.2). Each specific objective also affects decisions taken. For example, emphasizing the economy would shift decisions from a full closure to reopen and vaccinate while everyone can move freely. When the economy is emphasized, we notice a massive disease spread. If a government only aims to maintain a healthy economy, then the number of individuals, who are hospitalized, in critical condition, and dead sharply increases. Among each part of the reward function, although emphasizing the total infections or the death toll reduces the spread in each compartment, emphasizing the portion of healthy and vaccinated individuals seems to show the best situation with respect to the total infections and death toll. Between the total infections and the death toll, emphasizing the total infections seems to be more beneficial because of the transmission mechanism. Emphasizing the death rate does not directly affect infections; therefore, the higher spread still contributes to more deaths.

Table 5.3 Comparison of different tuning parameters in the reward function for age-based vaccination.

Compartment	**Economy**	**Total Infections**	**Death Toll**	**Healthy or Vaccinated**
$(\lambda, \mu, \rho, \pi)$	(5,1,1,1)	(1,5,1,1)	(1,1,5,1)	(1,1,1,5)
Infected	6,762,388	2,078,877	2,348,258	1,057,088
Hospitalized	385,494	102,179	127,259	66,881
ICU	119,828	28,796	37,156	21,365
Dead	39,014	10,218	11,147	8,360

Vaccine Decisions and Distributions Vaccination restrictions have been among important discussions during the pandemic. There is definitely good in giving priority to individuals with preexisting conditions or older people who might be more endangered from the pandemic. But older ages are among individuals who have the least amount of contact during the day. Therefore, their contribution to the spread is generally low. Hence, we want to analyze the trade-off between age-based vaccination and random vaccination. In random vaccination, we do not prioritize super-spreaders to vaccinate, but we allow them and the riskier categories to get vaccinated with the same probability. Comparing our **AVM** and **RVM** models can give us insight into the benefits of each. We notice from comparing Figures 5.11 and 5.12 that the **RVM** strategy offers a faster closure, and earlier vaccination start, hence also improving the economy. In another situation where a government tends to be cautious toward the total number of deaths, they can give higher weight to the ρ parameter, which represents the weight of the death toll. This would cause lower rewards when death rates increase; thus, the DRL agent will optimize the decision while focusing on minimizing the death rates. In addition, Figure 5.14 shows a comparison between the age-based and random models. Allowing super-spreaders to get vaccinated as early as possible during an outbreak reduces infections in general, hence explains the suggested faster reopening and better economic performance.

Table 5.4 Comparison of Vaccine Distribution Among Different Age Groups and Compartmental Values for Each Model

Subgroup	**NJ CDC Data**	**RVM**	**AVM**	**pdiff**^{1}	**pdif**^{2}
Vaccination under 50 (%)	45	67	55	-18	-33
Vaccination 50 to 75 (%)	79	66	94	42	20
Vaccination over 75 (%)	78	63	96	51	22
Infected	851,485	675,310	892,672	32	26
Hospitalized	668,201	652,431	665,185	2	2
ICU	153,657	142,122	153,268	8	8
Recovered	828,283	845,298	898,803	6	-2
Dead	24,702	25,080	24,151	-4	-2

Table 4.1 compares the vaccination percentage and compartmental statistics for random vaccination and age-based vaccination strategies, and real data. To clarify, the NJ CDC Data and AVM columns do not report data from the validation experiments. The NJ CDC Data is obtained from the CDC and AVM represents the results obtained from the DRL agent suggestions. Column **pdiff**[1] represents the percentage difference between the random vaccination and age-based vaccination models, while **pdiff**[2] calculates the percentage difference between the random vaccination data and CDC reported data for NJ. Notice that the vaccine distribution for the age groups differs between the two methods. Random vaccination slightly suggests that some portion of the younger people should get vaccinated as soon as vaccines are available, while the age-based model vaccinates almost all older age groups first. Compartmental data shows that the total number of infected individuals reduces by 32% when using a random vaccination strategy over the age-based vaccination strategy. Due to this, the number of the hospitalized and critical cases and recovered individuals slightly reduces as well. The total number of dead individuals though is slightly increased by 4%. This shows that random vaccination strategy can offer earlier reopening and slower spread, but fast reopening and not focusing on age-based vaccination strategy comes with a cost. Figure 4.1 also compares the random vaccination strategy with the real data reported from the CDC for NJ. We notice a similar trend to that of the age-based vaccination. The experiments show that random vaccination could reduce the number of infections but still reports a 2% higher death rate than that of the CDC statistics for NJ.

5.7 Discussion and Future Work

We present a Simulation-Deep Reinforcement Learning (SiRL) framework for epidemic decision-making. Our study shows that an agent can be trained to take actions based on different available interventions and epidemic infection situations. Our results show that more could be done in handling the COVID-19 pandemic spread in the US. Our DRL agent identifies situations that government agencies should have acted faster toward slowing the spread of the virus. In addition, we compare different vaccination strategies and provide insights on the trade-off between random and age-based vaccination strategies.

Future directions can include extensions from the DRL and agent-based simulation as well. Further experiments could tell us more if we consider racial or geographical data. The simulation model can be extended to account for additional costs or an economic value of a current infestation, including the interventions active at a point in time. Another DRL algorithm could be used to study how that changes agent performance. Finally, to study the flexibility of the framework other epidemic data in different regions of the U.S. and the world can be validated.

CHAPTER 6

SUMMARY AND FUTURE RESEARCH DIRECTIONS

6.1 Summary of Contributions

In this dissertation, we tackle the problem of biological infections in the forest and human species. We present novel methodologies, improvements to solution approaches, and recommendations to resource managers and decision-makers.

In Chapter 2, we presented an MSS-MIP formulation covering 472 km^2. We introduced history-dependent realization probabilities and a distance-dependent estimation spread covering an area of four 1-km distances from a specific site. We support decision-makers by visualizing where to treat, survey, and remove ash trees. In addition, we also provide insight to resource managers for different survey methods and their effectiveness. In Chapter 3, we derived a nested risk measure for a maximization problem and integrated it into a scenario-based formulation of an MSS-MIP problem. As a solution methodology, we adapted the scenario dominance cutting planes to a case of decision-dependent uncertainty. We support resource managers with a solution where they can observe the trade-off between following a risk-averse or risk-neutral approach.

In Chapter 4, we present a novel methodology to tackle zero-one integer problems with resource constraints. Our methodology is particularly applied using the multi-dimensional knapsack problem, which is at the root of many practical problems. Our framework is data-driven, where preprocessing is done first, and then the information extracted is feed to a deep reinforcement learning agent. For preprocessing, we develop a heuristic that reduces the dimensionality of the problem and estimates an item's worthiness value which is later used to sort items. To simplify the problem at hand and define training objectives, we develop an unsupervised learning algorithm using K-means to find a reasonable starting solution. Then, we formulate a 2D DRL environment to hold instance information about the selected and not selected items where the agent travels and decides which items to select by considering constraints and the objective. Finally, we tailor an algorithm ensuring collaboration between our DRL solution and CPLEX solver resulting in a powerful combination.

In Chapter 5, extend further our DRL framework to be used in epidemic control planning. To do so, we extend the agent-based simulation introduced in Kerr et al. [2020] with vaccinations of different strategies, the flexibility of a periodic active intervention definition, and other vaccination compartments. We introduce an age-based vaccination strategy for vaccines approved in the U.S. under the EUA by the FDA. We provide an external decision-maker to change the interventions measure in a preset frequency. At the end of every decision period, the simulation feeds compartmental statistics to the training network. We introduce a multi-objective reward objective where a decision-maker can make a trade-off between improving the economy and reducing the total infections and the death toll, or keeping people healthy for as long as possible.

6.2 Future Research Directions

Each of the studies presented in this dissertation can be further advanced by improving the proposed methodology and extending it to tackle similar applications. For example, a possible extension of the invasive species management model is the application to other invasive species that share similar age-structured biological properties. From the methodological point of view, deciding which cuts are more effective in the scenario dominance cutting planes is critical. Randomly selecting cutting planes or including all cuts in the problem solution might not be very effective. Other methods can be developed to select and add cutting planes more efficiently.

To better express the biological properties of the invaders, the long-distance dispersal mechanism could be extended.

Our DLR formulation can be extended to solve other classes of combinatorial optimization problems. Methodologically, improvements and more tuning can be made at any point of the DRL framework. For example, another way can be used to estimate the item's worthiness. A more accurate or faster method can be used to get an initial solution for the instances. Any improvements in both of the above suggestions will lead to an improvement in solution time and its accuracy. Focusing on the DRL environment, other formulations can also be useful. For example, extending to a 3D environment and increasing the action space as well could be a better way as the agent at each decision point will have more available actions. A new DRL algorithm tailored specifically for COPs would definitely improve the framework.

Future directions with respect to epidemic control using deep reinforcement include extensions of the agent-based simulation, deep reinforcement learning algorithm, and epidemic application. The agent-based simulation can be extended to involve more characteristics of the population, such as racial or geographical data. Other DRL algorithms can be tested or developed to study whether agents are trained faster or provide more accurate suggestions. Finally, other epidemic diseases can be validated and then tested using our agent-based simulation DRL framework.

APPENDIX A

FURTHER NOTES CHAPTER 2

Infestation Layer: surrounding neighbors with the same distance from a site

Infestation Level: classification of the severity of host infestation

Realization: a specific outcome regarding the degree of uncertain infestation after surveillance

Scenario: a combination of realizations for each time period and surveillance decisions

Site: a 1-km^2 area populated by ash trees

Surveillance Efficiency: the proportion of infested trees that are identified after surveillance

Stage: a time period in the stochastic scenario tree

Transition: change of the host infestation levels from one time period into another

Transition Population: estimated population of host trees without taking the maximum host population into account

A.1 Two-Stage Example of Possible Scenarios

Table A.1 presents all possible scenarios for a 2-stage problem. As an example of a two-stage problem, at period $t = 1$, we can decide whether to survey or not. If the site is surveyed, two possible infestation levels can be uncovered - a higher than expected (H) or lower than expected infestation (L). If no survey is applied, we assume a default medium infestation level (M) and use this value to compute the probabilities of spread. Thus, at period $t = 1$, three possible realizations are possible, as demonstrated by red, green, and yellow arcs (Table 2.3). In period 2, each square node generated at the end of the first period depicts the decisions to survey a site in that period. Similarly, if the site is surveyed, high or low infestation levels can be detected. If no surveillance is performed, we assume a default medium infestation level and use this value to update the probabilities of infestation at the next time period and so on.

Table A.1 All Possible Scenarios for a Two-Stage Formulation

Scenario Number	**Realization**	**Surveillance Regime**
1	H-H	Survey-Survey
2	H-L	Survey-Survey
3	H-M	Survey-Do Not Survey
4	L-H	Survey-Survey
5	L-L	Survey-Survey
6	L-M	Survey-Do Not Survey
7	M-H	Do Not Survey-Survey
8	M-L	Do Not Survey-Survey
9	M-M	Do Not Survey-Do Not Survey

A.2 Scenario Probability Algorithm

Algorithm 5 dynamically updates infestation realization probabilities in each period. We start by defining scnProb, which is an array that will hold the probability of encountering a certain infestation realization for a scenario ω at time t, where p_H holding the probability of encountering a high infestation realization and p_L representing a low infestation realization probability. Each realization will be represented by the variable Real, which is either equal to High(H) or Low(L) or Medium(M). We keep track of the probabilities for different infestation realizations at each time stage using variables p_H and p_L. We assign an equal probability for both high and low infestation realizations in the initial time stage. We update the probabilities as we traverse through all the time periods under each scenario. Whenever we encounter a pattern of repeating realizations, we update the probabilities of uncertain outcomes in the favor of the repeating event. At the end of the procedure, we obtain a two-dimensional set that lists the occurrence probabilities of infestation realizations at time t for each scenario ω (a $scnProb$ array). We then use the function $multiplyList(\omega_i)$ to multiply all probabilities in array ω_i, which gives the realizations of the infestation for each time t under a specific scenario ω with index $i = 1, \ldots, 3^t$. Therefore, after we multiply the probabilities in ω_i returning only one probability value for each scenario ω, we use the $normalizeProb(\omega_i)$ method to normalize the probability of each scenario ω.

Algorithm 5 Calculate probabilities for each scenario

```
Procedure: Probability Distribution
DEFINE $sncProb,\ p_H,\ p_L$
INITIALIZE $scnProb = \emptyset,\ p_H = 0.5,\ p_L = 0.5$
for each scenario $\omega \in \Omega$ do
    if $Real_t$ is High or Low or Medium then
        append $p_H,\ p_L\ or\ 1$ , respectively
    end if
    for each $Real$ in time period $t \in T$ do
        if $Real_t = Real_{t+1}$ and $Real_{t+1} =$ High then
            $p_H+ = 0.1$
            $p_L- = 0.1$
            if $Real_t$ is High or Low or Medium then
                append $p_H,\ p_L\ or\ 1$ , respectively
            end if
        else if $Real_t = Real_{t+1}$ and $Real_{t+1} =$ Low then
            $p_H- = 0.1$
            $p_L+ = 0.1$
            if $Real_t$ is High or Low or Medium then
                append $p_H,\ p_L\ or\ 1$ , respectively
            end if
        else
            if $Real_t$ is High or Low or Medium then
                append $p_H,\ p_L\ or\ 1$ , respectively
            end if
        end if
    end for
    Procedure: Scenario Probability
    for each scenario array $\omega_i$, $i = 1, \ldots, 3^t$ do
        return multiplyList($\omega_i$)
    end for
    for each scenario probability $\omega_i$, $i = 1, \ldots, 3^t$ do
        return normalizeProbability($\omega_i$)
    end for
end for
```

APPENDIX B

FURTHER NOTES CHAPTER 3

B.1 Model Description and Assumptions

This section presents a verbal description of the risk-averse multi-stage stochastic mixed integer programming model of the surveillance, treatment, and removal planning for the EAB infestation, and the assumptions made in the mathematical model. The formulation is described in detail in a companion paper [Bushaj et al., 2021b].

The risk-averse multi-stage stochastic MIP in Equations [(3.22)–(3.44)] is applied in a spatial setting and includes the spatial dynamics of EAB and its host population. We divide the study area into sites or neighborhoods. The ash tree population in each site is divided into healthy trees that are susceptible to infestation and infested trees belonging to three classes (levels): asymptomatic trees, symptomatic trees, and dead trees. Asymptomatic trees represent the lowest infestation (level 1), followed by symptomatic trees with visible signs of infestation (level 2), and if no treatment or removal is applied, trees die (level 3). Infested trees are the source of EAB spread to susceptible trees in the same site and surrounding sites. Each year, infested trees transition to the next, more severe infestation level, and susceptible trees may become infested through EAB spread within and between neighboring sites. Management decisions, including insecticide treatment and tree removal, are prescribed for each site depending on the outcome of surveillance, which reports the number of trees by infestation class in each site. We assume that decisions to treat the infested trees can be only effective when applied to asymptomatic trees while symptomatic and dead trees may be removed.

The number of healthy trees that may be infested at a site is uncertain, and we used a probabilistic depiction of spread. In our case, the number of newly infested trees at a site is a random variable that depends on the number of EAB adults produced at a given site and neighboring sites. Surveillance is critical because it provides information about the actual infestation, which allows making decisions about ash treatment and removal. The realization of the uncertain infestation variable depends on the binary surveillance actions that are integrated into a multi-stage scenario tree [Kıbış et al., 2021]. To simplify our model, we make two key assumptions about surveillance: 1) if surveillance is undertaken in a given time period, all of the sites are surveyed, and the number of susceptible and infested trees in each level become known; and 2) treatment and removal decisions for infested trees in levels 1 and 2 can be made only in the periods in which surveillance is made, and the number of infested trees is known. Dead trees (infestation level 3) are known and may be removed depending on the budget at each period.

We choose to formulate a multi-stage stochastic MIP to capture the stochastic spatial dynamics of EAB, its host population, and the efficacy of surveillance and insecticide treatment. Features of this spatial-dynamic system are: 1) infested ash trees are asymptomatic in the early stage of infestation (i.e., one cannot tell they are infested without inspection); 2) the realization of the uncertainty depends on the surveillance (decision-dependent uncertainty); 3) infestation of susceptible ash trees in a given site is a stochastic process that depends on the infested status of trees in a site and neighboring sites; 4) chemical treatment is effective only on asymptomatic ash trees, which have low levels of infestation; and 5) surveillance is needed to identify asymptomatic trees. These features do not fit into the two-stage modeling formulations.

B.1.1 Scenario Tree Generation

The uncertainty in our case-study model is exogenous, which is also known as decision-dependent uncertainty, because the realization of the uncertain infestation depends on the binary surveillance actions that are integrated into a multi-stage scenario tree Kıbış et al. [2021]. Given the assumptions described above, a scenario tree is formed based on the surveillance decisions and the stochastic infestation outcomes. Each scenario depicts a sequence of possible surveillance decisions and infestation outcomes. The set of scenarios includes all possible sequences of surveys, and an optimal solution is found for each scenario (sequence of surveys and infestation outcomes). The optimal solution for a given scenario includes the optimal treatment and removal decisions in each site in each period in which surveillance takes place and contingent on the infestation outcome, which is revealed in the survey. Comparing the optimal solutions among all the scenarios, we can identify the best sequence of surveillance decisions over the horizon and compare alternative sequences.

Figure B.1 depicts an example scenario tree with the sequences of possible surveillance decisions and the stochastic infestation outcomes over time and is based on the model of Kıbış et al. [2021]. The scenario tree starts in period one and at each branch through time shows a realization of the uncertain infestation levels. Two types of nodes in the scenario tree indicate the surveillance (black circles) and no surveillance (empty circles) conditions. Square nodes represent treatment or removal decisions. Given the uncertainty about EAB spread, each decision has two possible outcomes: high or low realization of the uncertain level of infestation. The outcome without surveillance (identified by empty circles) yields the expected value of the uncertain infestation level. The notation on each arc p_t^H, p_t^L and p_t^M, stands for the probability of detecting the infestation at a high (H) or low (L) level, or the expected infestation level (M) in the absence of surveillance in period t, producing 3^t scenarios. We assume a medium infestation level as an expected outcome without surveillance

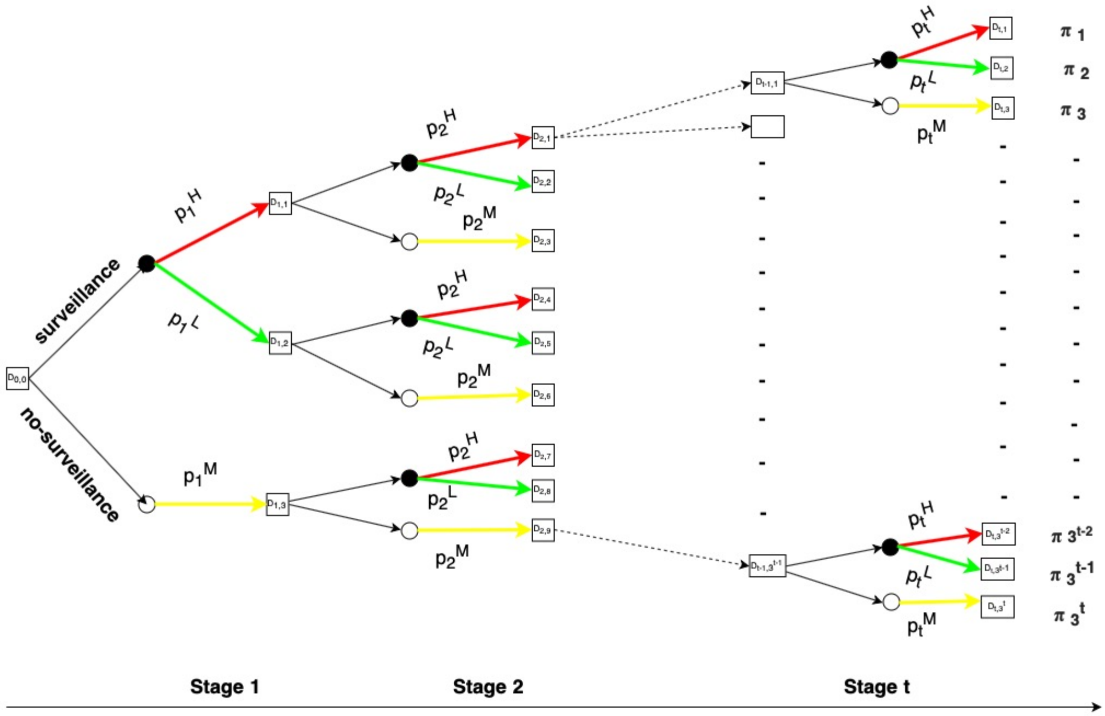


Figure B.1 Multi-stage scenario tree. Terms p_t^H and p_t^L denote the realization probabilities of high (H) and low (L) levels of infestation after surveillance; p_t^M denotes a default realization of medium infestation level (M) without surveillance. Black circles represent nodes with decisions after the surveillance; white circles depict nodes without surveillance; arcs leaving black and white circles depict possible realizations of the estimated beliefs about the number of susceptible and infested trees; red arrows depict realizations of high infestation levels; green arrows depict realizations of low infestation levels; yellow arrows depict anticipated levels of infestation without surveillance based on the initial belief. Squares $D_{t,s}$ depict treatment and removal decisions.
Source: Adapted from Kıbış et al. [2021], Bushaj et al. [2021b].

and use this level to update the number of infested trees at a given site without surveillance. We obtain an expected level of infestation by explicitly formulating a 4-km radius spread of infestation and spread probabilities from the infested site in the mathematical model. Thus, p_t^M represents this expected infestation level, while p_t^H and p_t^L represent an outcome of surveillance, which is higher or lower than the expected value, respectively. Terms $\pi_1 - \pi_{3t}$ denote the conditional probabilities of a scenario $\omega, \omega \in [1, 2, \ldots, 3^t]$. This probability is calculated by multiplying the probability of each realization through the whole scenario path and normalizing it over all scenarios, so the sum of the probabilities over all scenarios is equal to 1.

B.2 Time-Consistent Mean-Risk MSS-MIP

The general $\mathbb{E} - CVaR$ optimization problem that is formulated using the $CVaR_{\alpha^t}^{[\xi^t]}$ and linearization constraints (3.13) can be represented as a dynamic program shown below [Guo and Ryan, 2017]: For $t = 1$:

$$Z_{E-CVaR} = \max_{x^1, \eta^2} \begin{cases} f^1(x^1) + \eta^2 + \mathbb{E}_{\xi^2}\left[Q^2(x^1, \eta^2, \xi^{[2]})\right] \\ A^1 x^1 \leq b^1 \end{cases} \tag{B.1}$$

For $t = 2, \dots, T-1$:

$$Q^t(x^{t-1}, \eta^t, \xi^{[t]}) = \max_{x^t, \eta^{t+1}, v^t} \begin{cases} \eta^{t+1} - \frac{1}{\alpha^t} v^t + \mathbb{E}_{\xi^{t+1}}\left[Q^{t+1}(x^t, \eta^{t+1}, \xi^{[t+1]})\right] \\ A^t\left(\xi^{[t-1]}\right) x^{t-1}\left(\xi^{[t-1]}\right) + W^t\left(\xi^{[t]}\right) x^t\left(\xi^{[t]}\right) \leq b^t\left(\xi^{[t]}\right) \\ \eta^t - \sum_{t'=1}^{t} f^{t'}(x_\omega^{t'}, \xi_\omega^{t'}) \leq v^t \\ v^t \geq 0 \end{cases}$$

For $t = T$:

$$Q^T(x^{T-1}, \eta^T, \xi^{[T]}) = \max_{x^T, v^T} \begin{cases} -\frac{1}{\alpha^T} v^T \\ A^T\left(\xi^{[T-1]}\right) x^{T-1}\left(\xi^{[T-1]}\right) + W^T\left(\xi^{[T]}\right) x^T\left(\xi^{[T]}\right) \leq b^T\left(\xi^{[T]}\right) \\ \eta^T - \sum_{t'=1}^{T} f^{t'}(x_\omega^{t'}, \xi_\omega^{t'}) \leq v^T \\ v^T \geq 0 \end{cases}$$

Considering a case with finitely many realizations ξ and corresponding probabilities p_ξ and replicating η^t and v^t for each ξ, the equivalent scenario formulation of the

$\mathbb{E}-CVaR$ optimization problem (B.1) can be written as:

$$Z_{\mathbb{E}-CVaR_\alpha} = \max_{\substack{x^1,\ldots,x^T \\ \eta^2,\ldots,\eta^T \\ v^2,\ldots,v^T}} \sum_{p_\xi \in \Xi} p(\xi) \left[f^1 x^1 + \eta^2(\xi) - \frac{1}{\alpha^2} v^2(\xi) + \ldots + \eta^T(\xi) - \frac{1}{\alpha^T} v^T(\xi) \right] \tag{B.2}$$

subject to:

$$\begin{array}{lcll}
A^1 x^1 & \leq & b^1, & \\
A^t(\xi) x^{t-1}(\xi) + W^t(\xi) x^t(\xi) & \leq & b^t(\xi) & \forall\, \xi \in \Xi \; \forall\, t = 2, \ldots, T \\
\eta^t(\xi) - \sum_{t'=1}^{t} f^{t'}(x_\omega^{t'}, \xi_\omega^{t'}) & \leq & v^t(\xi) & \forall\, \xi \in \Xi \; \forall\, t = 2, \ldots, T \\
x^t(\xi) - \hat{x}^t(\xi^{[t]}) \quad \& \quad v^t(\xi) - \hat{v}^t(\xi^{[t]}) & = & 0 & \forall\, \xi \in \Xi \; \forall\, t = 1, \ldots, T \\
x^t(\xi),\; \hat{x}^t(\xi^{[t]}),\; v^t(\xi),\; \hat{v}^t(\xi^{[t]}) \in \mathbb{R}_+^{n^t - q^t} \times \mathbb{Z}_+^{q^t} & & & \forall\, \xi \in \Xi \; \forall\, t = 1, \ldots, T.
\end{array}$$

where $\hat{x}^t(\xi^{[t]})$ is the non-anticipative decision made at scenario realization ξ up to time t.

B.3 Case Study Data

We apply our risk-averse model to the case of EAB management in the state of New Jersey. We describe the economic and EAB-related biological data as well as the generation of test instances for the state of New Jersey in this section.

To obtain test cases as close as possible to the real infestation situation of New Jersey, we refer to the currently available infestation information from New Jersey (N.J.) Forest Services [Wilson et al., 2013] and enhance it based on some assumptions. Combining this data with the official detection year of each county provided from N.J. Forest Services [USDA, 2018], we run a four-nearest neighbor dispersal algorithm assuming two initial infestation cases, 1% and 2.5% of the total trees in a site, are infested. For each initial infestation, we consider three EAB dispersal rates. The base dispersal rate is inherited from Bushaj et al. [2021b], where the dispersal rates are calculated based on the observed data from experts. As the symptomatic trees are usually more infested with EAB galleries, we assume that they contribute more to the new infestation of susceptible trees [Knight et al., 2012]. In addition, we assume that dead trees do not have an impact on new infestations. To account for the effect of environmental and climate change in the EAB spread of N.J., we also assume infestation dispersal rates of 20% less and 20% more than the base case. We run a dispersal algorithm to simulate the dispersal of the EAB from the year an infestation was noticed until the current year (2020). We run the algorithm for each combined pair of data values for the initial infestation and the dispersal rate. In addition to the initial infestation and infestation dispersal assumptions, we use five different budgets for each data set. The budget levels are set based on the initial infestation. For the lower initial infestation, budgets vary from \$120,000 to \$300,000. For the higher initial infestation, the budget range from \$600,000 to \$2M. Combining two initial infestations, three infestation dispersal rates, and five budgets, we have a data set with 30 distinct instances. These data files represent the number of trees in each infestation level upon which we will run our model for the next five years to provide managerial insights.

Management activities to slow the spread of EAB include surveillance to detect the presence of EAB in ash trees, insecticide treatment of ash trees that are healthy or in the earliest stage of infestation, and removal of infested trees. Based on our discussion with N.J. Forest Service, we assume a surveillance method using traps followed by either treatment or removal of ash trees in the area of EAB detection. In our experiments, we assume the use of sticky traps, which have a likelihood to detect EAB if it is present on a tree of 0.5 [Ryall, 2015]. The surveillance cost for each sticky trap is estimated to be \$140, which is a 60% increase in the surveillance cost estimated in Bushaj et al. [2021b] for a case study of EAB management in Canada. Following surveillance, insecticide treatment of healthy or newly infested ash trees kills any EAB larvae present and prevents further infestation of those trees for two years. Removing infested ash trees prevents the spread of EAB adults but also reduces ash tree cover and associated benefits. We estimate the costs of insecticide treatment and removal using a survey of firms that provide treatment and removal services for ash trees in N.J. Treatment cost varies from \$13 to \$15 per diameter inch of the tree. On the information gathered from Wilson et al. [2013], we deduce that the average diameter is 14 inches. Therefore, on average, the treatment cost is estimated to be \$200 per tree. Removal cost varies a lot more because it does not depend solely on the tree diameter but also other factors, such as the location, difficulty to reach, and the height of the tree. Based on the surveys, the average cost to remove a tree is \$1280, which is about 60% more than the removal cost in the Canadian case considered in Bushaj et al. [2021b]. In accordance with the increase in removal cost, the monetary value of the benefits provided by an ash tree is estimated to be \$120 per tree compared to \$72 per tree in Bushaj et al. [2021b].

Figure B.2 shows a color map of each site's total population based on the data gathered in Wilson et al. [2013]. This color map refers only to the counties in the N.J. map that are not in gray shown in Figure B.3.

Figure B.3 shows the initial map with data provided by the U.S. Department of Agriculture (USDA) for the state of New Jersey. They report the year of initial detection for each county. For example, a data file is compromised of the ash tree population given in Figure B.2 and the infestation detection information presented in Figure B.3 with the initial infestation calculated by a 4-level dispersal algorithm executed for the number of years passed from the first detection.

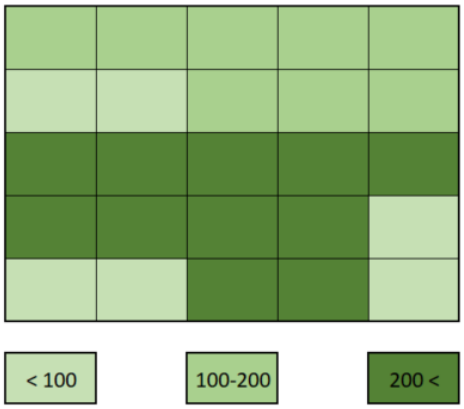


Figure B.2 New Jersey ash tree population

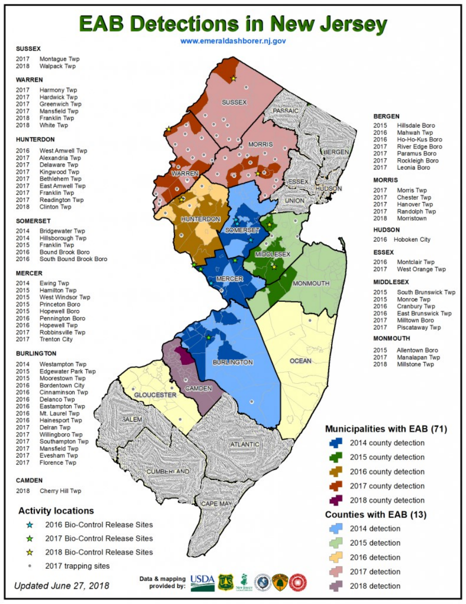


Figure B.3 New Jersey EAB detection data [USDA, 2018]

B.4 Gap Improvement Calculation for Sdc and Ssdc Cuts

In this section, we present linear programming relaxation solutions and integrality gap improvements of sdc and ssdc with respect to cplex (cpx). Table B.1 provides the following information for each column representing a particular instance defined by **II** (initial infestation size of the area [small (s) at 1% or large (l) at 2.5% infestation of the total ash trees]) and **DR** (infestation dispersal rate [slow (s), medium (m), fast (f)]), and the average in each row:

- **Obj**: Best objective value by default cpx;
- **Obj**^{LP_1}: Objective value of the initial problem relaxation before adding cuts;
- **Obj**^{LP_2}: Objective value of the relaxation after adding **sdc** cuts;
- **Obj**^{LP_3}: Objective value of the relaxation after adding **ssdc** cuts;
- **TimeLP**: CPU time required to solve the relaxation problem for **cpx**;
- **InitGap**(%):Percentage integrality gap of the formulation before inequalities are added ($InitGap = 100 \times (Obj - relaxObj)/Obj$), where $relaxObj$ and $bestObj$ are objective function values of the initial **cpx** LP relaxation and the best feasible solution, respectively;

- **RootGap**^{1}(%): Percentage integrality gap of the formulation after **sdc** inequalities are added ($RootGap = 100 \times (Obj - rootObj)/Obj$), where $rootObj$ is the objective function value of the LP relaxation after the **sdc** cuts are added;
- **RootGap**^{2}(%): Percentage integrality gap of the formulation after **ssdc** inequalities are added ($RootGap = 100 \times (Obj - rootObj)/Obj$), where $rootObj$ is the objective function value of the LP relaxation after the **ssdc** cuts are added;
- **GapImp**^{1}(%): Percentage improvement in the integrality gap at the root node for **sdc** ($GapImp = 100 \times (1 - rootObj/relaxObj)$; and
- **GapImp**^{2}(%): Percentage improvement in the integrality gap at the root node for **ssdc** ($GapImp = 100 \times (1 - rootObj/relaxObj)$.

Table B.1 Calculations for Integrality Gap Improvement by SDC and SSDC

(II,DR)	**(s,s)**	**(s,m)**	**(s,f)**	**(l,s)**	**(l,m)**	**(l,f)**	**Average**
Obj	2,150,592	1,192,578	265,106	420,554	196,576	-490,969	**622,406**
Obj^{LP_1}	2,162,000	1,201,871	267,656	423,139	198,471	-486,972	**627,694**
Obj^{LP_2}	2,159,862	1,199,208	267,224	422,946	198,122	-488,023	**626,556**
Obj^{LP_3}	2,157,869	1,197,958	267,132	422,755	198,221	-488,132	**625,967**
TimeLP	88.2	102	235.6	91.7	533.2	550.3	**266.8**
InitGap(%)	0.53	0.78	0.96	0.61	0.97	0.81	**0.78**
RootGap^1(%)	0.43	0.56	0.80	0.57	0.79	0.60	**0.62**
RootGap^2(%)	0.34	0.45	0.76	0.52	0.83	0.58	**0.58**
GapImp^1(%)	18.74	28.66	16.94	7.47	18.38	26.29	**19.78**
GapImp^2(%)	36.21	42.11	20.55	14.85	13.69	29.02	**25.24**

As seen in Table B.1, both sdc and ssdc help to improve the LP relaxation solution, the root gap, and the optimality gap compared to cpx.

Table B.2 Calculations for Integrality Gap Improvement by SDC and SSDC

(II,DR)	**Obj**	**TimeLP**	**InitGap**	**RootGap**^{1}	**RootGap**^{2}	**GapImp**^{1}(%)	**GapImp**^{2}(%)
(s,s)	2,150,592	88.2	0.53	0.43	0.34	18.74	36.21
(s,m)	1,192,578	102	0.78	0.56	0.45	28.66	42.11
(s,f)	265,106	235.6	0.96	0.8	0.76	16.94	20.55
(l,s)	420,554	91.7	0.61	0.57	0.52	7.47	14.85
(l,m)	196,576	533.2	0.97	0.79	0.83	18.38	13.69
(l,f)	-490,969	550.3	0.81	0.6	0.58	26.29	29.02
Average	**622,406**	**266.8**	**0.78**	**0.62**	**0.58**	**19.78**	**25.24**

B.5 Effect of Risk Parameters on Time Complexity

To study the impact of the risk parameter in the time complexity of the model, we provide results for a combination of three distinct α values and five distinct levels of λ. Table B.3 shows the CPU time required to solve each instance with a different combination of risk parameters. The data is averaged over five runs done for each budget level on three different data files [the data files with (1,1), (2,1), and (2,3) as presented on Table 3.1]. Here, we observe how the sdc and ssdc perform compared to cpx for the risk-neutral model and the risk-averse models with different combinations of risk parameters. In general, we notice a pattern of time increase from risk-neutral to risk-averse models. In the risk-averse models, on average, we notice that solution time changes slightly as we change α and λ, but the change is not uniform. Instances solved for $\alpha = 0.25$ prove to be the hardest for all risk-averse models, while for $\alpha = 0.5$ and $\alpha = 0.05$, the solution time does not differ much.

The increase in solution times regarding cpx from risk-neutral to risk-averse is obvious. However, when solving the sdc and ssdc models, the solution time is quite close to the risk-neutral formulation. This shows that the sdc and ssdc cuts still improve on risk-averse models for various values of the risk parameters with a similar solution time as the risk-neutral model.

Table B.3 Time Complexity of Risk-Neutral, CPX, SDC, and SSDC Models for Different Risk Parameters

$\lambda\backslash\alpha$	**Risk-Neutral**			**0.05**			**0.25**			**0.5**		
	cpx	**sdc**	**ssdc**	**cpx**	**sdc**	**ssdc**	**cpx**	**sdc**	**ssdc**	**cpx**	**sdc**	**ssdc**
0.001	218	144	146	249	129	127	286	132	141	275	138	131
0.1	218	144	146	255	131	147	269	145	135	280	143	131
1	218	144	146	273	136	144	305	147	148	276	139	149
10	218	144	146	352	230	238	393	246	254	355	199	229
1000	218	144	146	153	112	123	153	111	119	159	116	111
Total	1090	720	730	1283	738	780	1407	783	799	1345	736	752
Average	**218**	**144**	**146**	**257**	**147**	**156**	**282**	**156**	**159**	**269**	**147**	**150**

APPENDIX C

FURTHER NOTES ON CHAPTER 5

C.1 Notations and Abbreviations

Compartments

S	Susceptible (Healthy) individuals
V_1	Individuals vaccinated with the first shot
V_2	Individuals vaccinated with the second shot
E	Exposed but not yet infections
M	Infected individuals with mild symptoms
H	Infected individuals who were hospitalized
C	Infected individuals who were admitted in an Intensive Care Unit
I	Infected individuals of all severity (Mild + Hospitalized + ICU)
R	Recovered individuals
T	Tested individuals
Q	Quarantined individuals
D	Individuals who lost their life due to COVID-19

Notations

$\mathcal{X}$	Set of all actions $\mathcal{X} = 0, 1, \ldots, 9$
$\mathcal{J}$	Set of all simulation period in a simulation $\mathcal{J} = 1, 2, \ldots, J$
s	Step size of the simulation (in days)
α	Weight of I in Equation 5.1
β	Weight of H in Equation 5.1
γ	Weight of C in Equation 5.1
λ	Weight of E_t in Equation 5.2
μ	Weight of I in Equation 5.2
ρ	Weight of D in Equation 5.2
π	Weight of $S + V1 + V2$ in Equation 5.2
Δ	Total population used in the simulation
σ	Total simulation periods
Θ	Compartmental statistics
θ	A state made up of compartmental statistics
Ω	Trained DQN Model

$|S - S_s|$: Absolute difference between the real healthy proportion of the population (S) and simulated healthy proportion of the population (S_s)

$|I - I_s|$: Absolute difference between the real total infected proportion of the population (I) and simulated total infected proportion of the population (I_s)

$|H - H_s|$: Absolute difference between the real hospitalized proportion of the population (H) and simulated hospitalized proportion of the population (H_s)

$|C - C_s|$: Absolute difference between the real ICU proportion of the population (C) and simulated ICU proportion of the population (C_s)

$|D - D_s|$: Absolute difference between the real dead proportion of the population (D) and simulated dead proportion of the population (D_s)

$|R - R_s|$: Absolute difference between the real recovered proportion of the population (R) and simulated recovered proportion of the population (R_s)

$|T - T_s|$: Absolute difference between the real treated proportion of the population (T) and simulated treated proportion of the population (T_s)

$|V_1 - V_{1s}|$: Absolute difference between the real first-shot vaccinated proportion of the population (V_1) and simulated first-shot vaccinated proportion of the population (V_{1s})

$|V_2 - V_{2s}|$: Absolute difference between the real two-shot vaccinated proportion of the population (V_2) and simulated two-shot vaccinated proportion of the population (V_{2s})

REFERENCES

Daniel A Herms and Deborah Mccullough. Emerald ash borer invasion of north america: History, biology, ecology, impacts, and management. *Annual Review of Entomology*, 59, 10 2013. doi: 10.1146/annurev-ento-011613-162051.

Fouad Ben Abdelaziz, Belaid Aouni, and Rimeh El Fayedh. Multi-objective stochastic programming for portfolio selection. *European Journal of Operational Research*, 177(3):1811–1823, 2007.

Carlo Acerbi and Dirk Tasche. Expected shortfall: a natural coherent alternative to value at risk. *Economic Notes*, 31(2):379–388, 2002.

Reza Refaei Afshar, Yingqian Zhang, Murat Firat, and Uzay Kaymak. A state aggregation approach for solving knapsack problem with deep reinforcement learning. In *Asian Conference on Machine Learning*, pages 81–96, 2020.

Shabbir Ahmed. Convexity and decomposition of mean-risk stochastic programs. *Mathematical Programming*, 106(3):433–446, 2006.

Heidi J Albers, Carolyn Fischer, and James N Sanchirico. Invasive species management in a spatially heterogeneous world: Effects of uniform policies. *Resource and Energy Economics*, 32(4):483–499, 2010.

Douglas Alem and Reinaldo Morabito. Risk-averse two-stage stochastic programs in furniture plants. *Operations Research Spectrum*, 35, 11 2013. doi: 10.1007/s00291-012-0312-5.

Antonio Alonso-Ayuso, Laureano F Escudero, Monique Guignard, and Andres Weintraub. Risk management for forestry planning under uncertainty in demand and prices. *European Journal of Operational Research*, 267(3): 1051–1074, 2018.

Amal Adel Alzu'bi, Sanaa Ibrahim Abu Alasal, and Valerie JM Watzlaf. A simulation study of coronavirus as an epidemic disease using agent-based modeling. *Perspectives in Health Information Management*, 18(Winter), 2021.

Philippe Artzner. Thinking coherently. *Risk*, pages 68–71, 1997.

Badar Nadeem Ashraf. Economic impact of government interventions during the covid-19 pandemic: International evidence from financial markets. *Journal of Behavioral and Experimental Finance*, 27:100371, 2020.

Juliann E Aukema, Brian Leung, Kent Kovacs, Corey Chivers, Kerry O Britton, Jeffrey Englin, Susan J Frankel, Robert G Haight, Thomas P Holmes, Andrew M Liebhold, et al. Economic impacts of non-native forest insects in the continental united states. *PLoS One*, 6(9), 2011.

Raghav Awasthi, Keerat Kaur Guliani, Saif Ahmad Khan, Aniket Vashishtha, Mehrab Singh Gill, Arshita Bhatt, Aditya Nagori, Aniket Gupta, Ponnurangam Kumaraguru, and Tavpritesh Sethi. Vacsim: Learning effective strategies for covid-19 vaccine distribution using reinforcement learning. *arXiv preprint arXiv:2009.06602*, 2020.

Egon Balas and Clarence H Martin. Pivot and complement–a heuristic for 0-1 programming. *Management Science*, 26(1):86–96, 1980. doi: 10.1287/mnsc.26.1.86. URL `https://doi.org/10.1287/mnsc.26.1.86`.

Stefan Balev, Nicola Yanev, Arnaud Fréville, and Rumen Andonov. A dynamic programming based reduction procedure for the multidimensional 0–1 knapsack problem. *European Journal of Operational Research*, 186(1):63–76, 2008.

Thomas Barrett, William Clements, Jakob Foerster, and Alex Lvovsky. Exploratory combinatorial optimization with reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3243–3250, Apr. 2020. doi: 10.1609/aaai.v34i04.5723. URL `https://ojs.aaai.org/index.php/AAAI/article/view/5723`.

Peter WJ Baxter and Hugh P Possingham. Optimizing search strategies for invasive pests: Learn before you leap. *Journal of Applied Ecology*, 48(1):86–95, 2011.

BBC. Coronavirus wipes out most of world's major sports events on an unprecedented day. `https://www.bbc.com/sport/51880582`, 2020. Accessed: 07-06-2021.

Bryan P Bednarski, Akash Deep Singh, and William M Jones. On collaborative reinforcement learning to optimize the redistribution of critical medical supplies throughout the COVID-19 pandemic. *Journal of the American Medical Informatics Association*, 28(4):874–878, 12 2020. ISSN 1527-974X. doi: 10.1093/jamia/ocaa324.

David NF Bell and David G Blanchflower. Us and uk labour markets before and during the covid-19 crash. *National Institute Economic Review*, 252:R52–R69, 2020.

Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. *Computing Research Repository*, abs/1611.09940, 2016. URL `http://arxiv.org/abs/1611.09940`.

Dimitris Bertsimas and Ramazan Demir. An approximate dynamic programming approach to multidimensional knapsack problems. *Management Science*, 48 (4):550–565, 2002.

Alyssa Bilinski, Joshua A Salomon, John Giardina, Andrea Ciaranello, and Meagan C Fitzpatrick. Passing the test: a model-based analysis of safe school-reopening strategies. *Annals of Internal Medicine*, 2021.

Alain Billionnet. Mathematical optimization ideas for biodiversity conservation. *European Journal of Operational Research*, 231(3):514–534, 2013.

John R Birge and Francois Louveaux. *Introduction to Stochastic Programming.* Springer Science and Business Media, Chester, UK, 2011.

Andrei Borshchev and Alexei Filippov. From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools. In *Proceedings of the 22nd international conference of the system dynamics society*, volume 22, pages 25–29. Citeseer, 2004.

Vincent Boyer, Moussa Elkihel, and Didier El Baz. Heuristics for the 0–1 multidimensional knapsack problem. *European Journal of Operational Research*, 199 (3):658–664, 2009.

Sally C Brailsford, Tillal Eldabi, Martin Kunc, Navonil Mustafee, and Andres F Osorio. Hybrid simulation modelling in operational research: A state-of-the-art review. *European Journal of Operational Research*, 278(3):721–737, 2019.

James Buck and Jordan Marshall. Hitchhiking as a secondary dispersal pathway for adult emerald ash borer, agrilus planipennis. *Great Lakes Entomologist*, 41: 155–157, 09 2008.

Sabah Bushaj and İ Esra Büyüktahtakın. A deep reinforcement learning approach for solving multi-dimensional knapsack problem. *Under Preparation*, 2021.

Sabah Bushaj, İ Esra Büyüktahtakın, and Robert G. Haight. Risk-averse multi-stage stochastic optimization for surveillance and operations planning of a forest insect infestation. *Under Review*, 2021a.

Sabah Bushaj, İ Esra Büyüktahtakın, Denys Yemshanov, and Robert G Haight. Optimizing surveillance and management of emerald ash borer in urban environments. *Natural Resource Modeling*, 34(1):e12267, 2021b.

I Esra Büyüktahtakın. Dynamic programming via linear programming. *Wiley Encyclopedia of Operations Research and Management Science. Wiley, Hoboken, NJ*, 2011.

İ Esra Büyüktahtakın. Stage-t scenario dominance for risk-averse multi-stage stochastic mixed-integer programs. *Submitted for Publication*, pages 1–36, 2020.

İ Esra Büyüktahtakın and Robert G Haight. A review of operations research models in invasive species management: State-of-the-art, challenges, and future directions. *Annals of Operations Research*, 271(2):357–403, 2018.

İ Esra Büyüktahtakın and Joseph C Hartman. A mixed-integer programming approach to the parallel replacement problem under technological change. *International Journal of Production Research*, 54(3):680–695, 2016.

İ Esra Büyüktahtakın and Ning Liu. Dynamic programming approximation algorithms for the capacitated lot-sizing problem. *Journal of Global Optimization*, 65(2):231–259, 2016.

İ Esra Büyüktahtakın, Zhuo Feng, George Frisvold, Ferenc Szidarovszky, and Aaryn Olsson. A dynamic model of controlling invasive species. *Computers and Mathematics with Applications*, 62(9):3326–3333, 2011.

I Esra Büyüktahtakin, Zhuo Feng, Aaryn D Olsson, George Frisvold, and Ferenc Szidarovszky. Invasive species control optimization as a dynamic spatial process: an application to buffelgrass (pennisetum ciliare) in arizona. *Invasive Plant Science and Management*, 7(1):132–146, 2014.

İ Esra Büyüktahtakın, Zhuo Feng, and Ferenc Szidarovszky. A multi-objective optimization approach for invasive species control. *Journal of the Operational Research Society*, 65(11):1625–1635, 2014a.

İ Esra Büyüktahtakın, J Cole Smith, Joseph C Hartman, and Shangyuan Luo. Parallel asset replacement problem under economies of scale with multiple challengers. *The Engineering Economist*, 59(4):237–258, 2014b.

İ Esra Büyüktahtakın, Eyyüb Y Kıbış, Halil I Cobuloglu, Gregory R Houseman, and J Tanner Lampe. An age-structured bio-economic model of invasive species management: Insights and strategies for optimal control. *Biological Invasions*, 17(9):2545–2563, 2015.

İ Esra Büyüktahtakın, Emmanuel des Bordes, and Eyyüb Y Kıbış. A new epidemics–logistics model: Insights into controlling the Ebola virus disease in West Africa. *European Journal of Operational Research*, 265(3):1046–1063, 2018a.

İ Esra Büyüktahtakın, J Cole Smith, and Joseph C Hartman. Partial objective inequalities for the multi-item capacitated lot-sizing problem. *Computers and Operations Research*, 91:132–144, 2018b.

Min Cai and Jianwen Luo. Influence of covid-19 on manufacturing industry and corresponding countermeasures from supply chain perspective. *Journal of Shanghai Jiaotong University (Science)*, 25(4):409–416, 2020.

David Cappaert, Deborah G McCullough, Therese M Poland, and Nathan W Siegert. Emerald ash borer in north america: a research and regulatory challenge. *American Entomologist. 51 (3): 152-165.*, 51(3), 2005.

Alberto Caprara, Hans Kellerer, Ulrich Pferschy, and David Pisinger. Approximation algorithms for knapsack problems with cardinality constraints. *European Journal of Operational Research*, 123(2):333–345, 2000. ISSN 0377-2217. doi: https://doi.org/10.1016/S0377-2217(99)00261-1. URL `https://www.sciencedirect.com/science/article/pii/S0377221799002611`.

Claus C CarøE and Rüdiger Schultz. Dual decomposition in stochastic integer programming. *Operations Research Letters*, 24(1-2):37–45, 1999.

Rabail Chaudhry, George Dranitsaris, Talha Mubashir, Justyna Bartoszko, and Sheila Riazi. A country level analysis measuring the impact of government actions, country preparedness and socioeconomic factors on covid-19 mortality and related health outcomes. *EClinicalMedicine*, 25:100464, 2020.

Jianyu Chen, Bodi Yuan, and Masayoshi Tomizuka. Model-free deep reinforcement learning for urban autonomous driving. In *2019 Institute of Electrical and Electronics Engineers Intelligent Transportation Systems Conference (ITSC)*, pages 2765–2771. Institute of Electrical and Electronics Engineers, 2019.

Weijia Chen, Yuedong Xu, and Xiaofeng Wu. Deep reinforcement learning for multi-resource multi-machine job scheduling. *arXiv preprint arXiv:1711.07440*, 2017.

Weihsueh A Chiu, Rebecca Fischer, and Martial L Ndeffo-Mbah. State-level needs for social distancing and contact tracing to contain covid-19 in the united states. *Nature Human Behaviour*, 4(10):1080–1090, 2020.

Paul C Chu and John E Beasley. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics*, 4(1):63–86, 1998.

Halil I Cobuloglu and İ Esra Büyüktahtakın. Food vs. biofuel: An optimization approach to the spatio-temporal analysis of land-use competition and environmental impacts. *Applied Energy*, 140:418–434, 2015a.

Halil I Cobuloglu and İ Esra Büyüktahtakın. A stochastic multi-criteria decision analysis for sustainable biomass crop selection. *Expert Systems with Applications*, 42(15-16):6065–6074, 2015b.

Halil I Cobuloglu and İ Esra Büyüktahtakın. A two-stage stochastic mixed-integer programming approach to the competition of biofuel and food production. *Computers and Industrial Engineering*, 107:251–263, 2017.

Halil Ibrahim Cobuloglu and İ Esra Büyüktahtakın. A mixed-integer optimization model for the economic and environmental analysis of biomass production. *Biomass and Bioenergy*, 67:8–23, 2014.

International Olympic Committee. Joint statement from the international olympic committee and the tokyo 2020 organising committee. `https://olympics.com/ioc/news/joint-statement-from-the-international-olympic-committee-and-the-tokyo-2020-organising-committee`, 2020. Accessed: 07-06-2021.

Michael Common and Sigrid Stagl. *Ecological economics: an introduction*. Cambridge University Press, Sussex, UK, 2005.

Sebastian Contreras, Jonas Dehning, Matthias Loidolt, Johannes Zierenberg, F Paul Spitzner, Jorge H Urrea-Quintero, Sebastian B Mohr, Michael Wilczek, Michael Wibral, and Viola Priesemann. The challenges of containing sars-cov-2 via test-trace-and-isolate. *Nature communications*, 12(1):1–13, 2021.

Özlem Coşgun and İ Esra Büyüktahtakın. Stochastic dynamic resource allocation for hiv prevention and treatment: An approximate dynamic programming approach. *Computers and Industrial Engineering*, 118:423–439, 2018.

IBM ILOG Cplex. V12. 1: User's manual for cplex. *International Business Machines Corporation*, 46(53):157, 2009.

Hanjun Dai, Bo Dai, and Le Song. Discriminative embeddings of latent variable models for structured data. *Computing Research Repository*, abs/1603.05629, 2016. URL http://arxiv.org/abs/1603.05629.

Hanjun Dai, Elias B. Khalil, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. *Computing Research Repository*, abs/1704.01665, 2017. URL http://arxiv.org/abs/1704.01665.

Jennifer M Dan, Jose Mateus, Yu Kato, Kathryn M Hastie, Esther Dawen Yu, Caterina E Faliti, Alba Grifoni, Sydney I Ramirez, Sonya Haupt, April Frazier, et al. Immunological memory to sars-cov-2 assessed for up to 8 months after infection. *Science*, 371(6529), 2021.

Thomas K Dasaklis, Costas P Pappis, and Nikolaos P Rachaniotis. Epidemics control and logistics operations: A review. *International Journal of Production Economics*, 139(2):393–410, 2012.

Jan de Mooij, Davide Dell Anna, Parantapa Bhattacharya, Mehdi Dastani, Brian Logan, and Samarth Swarup. Quantifying the effects of norms on covid-19 cases using an agent-based simulation. In *Proceedings of the The 22nd International Workshop on Multi-Agent-Based Simulation (MABS)*, 2021.

Arthur Delarue, Ross Anderson, and Christian Tjandraatmadja. Reinforcement learning with combinatorial actions: An application to vehicle routing. *arXiv preprint arXiv:2010.12001*, 2020.

Emmanuel des Bordes and İ Esra Büyüktahtakın. Optimizing capital investments under technological change and deterioration: A case study on mri machine replacement. *The Engineering Economist*, 62(2):105–131, 2017.

Ryan D DeSantis, W Keith Moser, Robert J Huggett, Ruhong Li, David N Wear, and Patrick D Miles. Modeling the effects of emerald ash borer on forest composition in the midwest and northeast united states. 2013. doi: https://doi.org/10.2737/NRS-GTR-112.

Gregory Dobson. Worst-case analysis of greedy heuristics for integer programming with nonnegative data. *Mathematics of Operations Research*, 7(4):515–531, 1982.

Marco D'Orazio, Gabriele Bernardini, and Enrico Quagliarini. How to restart? an agent-based simulation model towards the definition of strategies for covid-19 "second phase" in public buildings, 2020.

Jian J Duan, Leah S Bauer, and Roy G Van Driesche. Emerald ash borer biocontrol in ash saplings: The potential for early stage recovery of north american ash trees. *Forest Ecology and Management*, 394:64–72, 2017.

Joan G. Ehrenfeld. Ecosystem consequences of biological invasions. *Annual Review of Ecology, Evolution, and Systematics*, 41:59–80, 2010.

Rebecca S Epanchin-Niell, Robert G Haight, Ludek Berec, John M Kean, and Andrew M Liebhold. Optimal surveillance and eradication of invasive species in heterogeneous landscapes. *Ecology Letters*, 15(8):803–812, 2012.

Joshua M Epstein. Modelling to contain pandemics. *Nature*, 460(7256):687–687, 2009.

Laureano F Escudero, María Araceli Garín, and Aitziber Unzueta. Scenario cluster lagrangean decomposition for risk averse in multistage stochastic optimization. *Computers and Operations Research*, 85:154–171, 2017.

Laureano F Escudero, M Araceli Garín, Juan F Monge, and Aitziber Unzueta. On preparedness resource allocation planning for natural disaster relief under endogenous uncertainty with time-consistent risk-averse management. *Computers and Operations Research*, 98:84–102, 2018a.

Laureano F Escudero, Juan Francisco Monge, and Dolores Romero Morales. On the time-consistent stochastic dominance risk averse measure for tactical supply chain planning under uncertainty. *Computers and Operations Research*, 100: 270–286, 2018b.

Laureano F Escudero, M Araceli Garín, Juan F Monge, and Aitziber Unzueta. Some matheuristic algorithms for multistage stochastic optimization models with endogenous uncertainty and risk management. *European Journal of Operational Research*, 285(3):988–1001, 2020.

Marc Etheve, Zacharie Alès, Côme Bissuel, Olivier Juan, and Safia Kedad-Sidhoum. Reinforcement learning for variable selection in a branch and bound algorithm. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 176–185, Vienna, AT, 2020. Springer.

Kyle Eyvindson and Zhuo Cheng. Implementing the conditional value at risk approach for even-flow forest management planning. *Canadian Journal of Forest Research*, 46(5):637–644, 2016.

George S Fishman. *Discrete-event simulation: modeling, programming, and analysis.* Springer Science and Business Media, Chester, UK, 2013.

Charles E Flower, Kathleen S Knight, Joanne Rebbeck, and Miquel A Gonzalez-Meler. The relationship between the emerald ash borer (agrilus planipennis) and ash (fraxinus spp.) tree decline: Using visual canopy condition assessments and leaf isotope measurements to assess pest damage. *Forest Ecology and Management*, 303:143–147, 2013.

G Edward Fox and Gary D Scudder. A heuristic with tie breaking for certain 0–1 integer programming models. *Naval Research Logistics Quarterly*, 32(4): 613–623, 1985.

Arnaud Fréville and Gérard Plateau. An exact search for the solution of the surrogate dual of the 0–1 bidimensional knapsack problem. *European Journal of Operational Research*, 68(3):413–421, 1993.

Alan M Frieze and Michael RB Clarke. Approximation algorithms for the m-dimensional 0–1 knapsack problem: Worst-case and probabilistic analyses. *European Journal of Operational Research*, 15(1):100–109, 1984. ISSN 0377-2217. doi: https://doi.org/10.1016/0377-2217(84)90053-5. URL `https://www.sciencedirect.com/science/article/pii/0377221784900535`.

Charis M Galanakis, Myrto Rizou, Turki MS Aldawoud, Ilknur Ucak, and Neil J Rowan. Innovations and technology disruptions in the food sector within the covid-19 pandemic and post-lockdown era. *Trends in Food Science and Technology*, 2021.

Belinda Gallardo, Miguel Clavero, Marta I Sánchez, and Montserrat Vilà. Global ecological impacts of invasive species in aquatic ecosystems. *Global Change Biology*, 22(1):151–163, 2016.

Dylan Gaspar, Yun Lu, Myung Soon Song, and Francis J Vasko. Simple population-based metaheuristics for the multiple demand multiple-choice multidimensional knapsack problem. *International Journal of Metaheuristics*, 7(4):330–351, 2020.

Bezalel Gavish and Hasan Pirkul. Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality. *Mathematical Programming*, 31(1): 78–105, 1985.

Bezalel Gavish and Hasan Pirkul. Computer and database location in distributed computer systems. *Institute of Electrical and Electronics Engineers Transactions on Computers*, 35(7):583–590, 1986.

Navid Mahdizadeh Gharakhanlou and Navid Hooshangi. Spatio-temporal simulation of the novel coronavirus (covid-19) outbreak using the agent-based modeling approach (case study: Urmia, iran). *Informatics in Medicine Unlocked*, 20: 100403, 2020.

Melissa Gillisa, Ahmed Saifa, Noreen Kamala, and Matthew Murphy. A simulation-optimization framework for optimizing response strategies to epidemics. *Submitted for Review*, 2021.

Osea Giuntella, Kelly Hyde, Silvia Saccardo, and Sally Sadoff. Lifestyle and mental health disruptions during covid-19. *Proceedings of the National Academy of Sciences*, 118(9), 2021.

Fred Glover and Gary A Kochenberger. Critical event tabu search for multidimensional knapsack problems. In *Meta-heuristics*, pages 407–427. Springer, Boston, MA, 1996.

Rajeev K Goel, James W Saunoris, and Srishti S Goel. Supply chain performance and economic growth: The impact of covid-19 disruptions. *Journal of Policy Modeling*, 43(2):298–316, 2021.

David Goldsman, Richard E Nance, and James R Wilson. A brief history of simulation revisited. In *Proceedings of the 2010 Winter Simulation Conference*, pages 567–574. Institute of Electrical and Electronics Engineers, 2010.

Ralf Gollmer, Frederike Neise, and Rüdiger Schultz. Stochastic programs with first-order dominance constraints induced by mixed-integer linear recourse. *Society for Industrial and Applied Mathematics Journal on Optimization*, 19(2):552–571, 2008.

Ralf Gollmer, Uwe Gotzes, and Rüdiger Schultz. A note on second-order stochastic dominance constraints induced by mixed-integer linear recourse. *Mathematical Programming*, 126(1):179–190, 2011.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press Cambridge, Cambridge, MA, 2016.

Richard S Gray and Mohammad Torshizi. Update to agriculture, transportation, and the covid-19 crisis. *Canadian Journal of Agricultural Economics*, 69(2): 281–289, 2021.

Gary J Griffin. Blight control and restoration of the american chestnut. *Journal of Forestry*, 98(2):22–27, 2000.

Shenshen Gu, Tao Hao, and Hanmei Yao. A pointer network based deep learning algorithm for unconstrained binary quadratic programming problem. *Neurocomputing*, 390:1 – 11, 2020. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2019.06.111. URL `http://www.sciencedirect.com/science/article/pii/S0925231220303398`.

Ge C Guo and Sarah M Ryan. Progressive hedging lower bounds for time consistent risk-averse multistage stochastic mixed-integer programs. *URL https://works.bepress. com/sarah_m_ryan/93*, 2017.

Rohit Gupta and Stephanie R Morain. Ethical allocation of future covid-19 vaccines. *Journal of Medical Ethics*, 47(3):137–141, 2021.

Robert A Haack. Exotic bark-and wood-boring coleoptera in the united states: recent establishments and interceptions. *Canadian Journal of Forest Research*, 36(2): 269–288, 2006.

Robert A Haack and Toby R Petrice. Emerald ash borer adult dispersal. In *In: Mastro, Victor; Reardon, Richard, comps. Emerald ash borer research and technology development meeting; 2003 September 30-October 1; Port Huron, MI. FHTET 2004-03. Morgantown, WV: US Forest Service, Forest Health Technology Enterprise Team: 10.*, 2003.

Robert A Haack, Franck Hérard, Jianghua Sun, and Jean J Turgeon. Managing invasive populations of asian longhorned beetle and citrus longhorned beetle: A worldwide perspective. *Annual Review of Entomology*, 55:521–546, 2010.

Said Hanafi and Arnaud Freville. An efficient tabu search approach for the 0–1 multidimensional knapsack problem. *European Journal of Operational Research*, 106(2-3):659–675, 1998.

Joseph C Hartman, İ Esra Büyüktahtakin, and J Cole Smith. Dynamic-programming-based inequalities for the capacitated lot-sizing problem. *Institute of Industrial Engineers Transactions*, 42(12):915–930, 2010.

Mostafa Hasan, İ Esra Büyüktahtakın, and Elshami Elamin. A multi-criteria ranking algorithm (mcra) for determining breast cancer therapy. *Omega*, 82:83–101, 2019.

Hado Hasselt. Double q-learning. *Advances in Neural Information Processing Systems*, 23:2613–2621, 2010.

Christian Haul and Stefan Voss. Using surrogate constraints in genetic algorithms for solving multidimensional knapsack problems. In *Advances in computational and stochastic optimization, logic programming, and heuristic search*, pages 235–251. Springer, Boston, MA, 1998.

Frederick S Hillier. Efficient heuristic procedures for integer linear programming with an interior. *Operations Research*, 17(4):600–637, 1969. doi: 10.1287/opre.17.4.600. URL `https://doi.org/10.1287/opre.17.4.600`.

Robert Hinch, William J M Probert, Anel Nurtay, Michelle Kendall, Chris Wymant, Matthew Hall, Katrina Lythgoe, Ana Bulas Cruz, Lele Zhao, Andrea Stewart, Luca Ferretti, Daniel Montero, James Warren, Nicole Mather, Matthew Abueg, Neo Wu, Anthony Finkelstein, David G Bonsall, Lucie Abeler-Dörner, and Christophe Fraser. Openabm-covid19 - an agent-based model for non-pharmaceutical interventions against covid-19 including contact tracing. *medRxiv*, 2020. doi: 10.1101/2020.09.16.20195925. URL `https://www.medrxiv.org/content/early/2020/09/22/2020.09.16.20195925`.

John Hof. Optimizing spatial and dynamic population-based control strategies for invading forest pests. *Natural Resource Modeling*, 11(3):197–216, 1998.

Elizabeth E Holmes, Mark A Lewis, John E Banks, and RR Veit. Partial differential equations in ecology: spatial interactions and population dynamics. *Ecology*, 75(1):17–29, 1994.

Tito Homem-de-Mello and Bernardo K Pagnoncelli. Risk aversion in multistage stochastic programming: A modeling and algorithmic perspective. *European Journal of Operational Research*, 249:188–199, 2016.

A Hopkin, Peter de Groot, and Jean J Turgeon. Alien forest insects: What's bugging us in ontario. *Emerald Ash Borer and Asian Longhorned beetle. Forest Health and Biodiversity News*, 8:1–2, 2004.

Tetsuya Horie, Robert G Haight, Frances R Homans, and Robert C Venette. Optimal strategies for the surveillance and control of forest pathogens: A case study with oak wilt. *Ecological Economics*, 86:78–85, 2013.

Haoyuan Hu, Xiaodong Zhang, Xiaowei Yan, Longfei Wang, and Yinghui Xu. Solving a new 3d bin packing problem with deep reinforcement learning method, 2017.

Yuh-Jong Hu and Shang-Jen Lin. Deep reinforcement learning for optimizing finance portfolio management. In *2019 Amity International Conference on Artificial Intelligence (AICAI)*, pages 14–20. Institute of Electrical and Electronics Engineers, 2019.

Christian D Hubbs, Hector D Perez, Owais Sarwar, Nikolaos V Sahinidis, Ignacio E Grossmann, and John M Wassick. Or-gym: A reinforcement learning library for operations research problem. *arXiv preprint arXiv:2008.06319*, 2020.

Ray G Huffaker, Mahadev G Bhat, and Suzanne M Lenhart. Optimal trapping strategies for diffusing nuisance-beaver populations. *Natural Resource Modeling*, 6(1):71–97, 1992.

David E Jennings, Jian J Duan, Dick Bean, Juli R Gould, Kimberly A Rice, and Paula M Shrewsbury. Monitoring the establishment and abundance of introduced parasitoids of emerald ash borer larvae in maryland, usa. *Biological Control*, 101:138–144, 2016.

Alistair EW Johnson, Mohammad M Ghassemi, Shamim Nemati, Katherine E Niehaus, David A Clifton, and Gari D Clifford. Machine learning and decision support in critical care. *Proceedings of the Institute of Electrical and Electronics Engineers*, 104(2):444–466, 2016.

Lora Jones, Daniele Palumbo, and David Brown. Coronavirus: How the pandemic has changed the world economy. `https://www.bbc.com/news/business-51706225`, 2021. Accessed: 07-06-2021.

Steven A Juliano and L Philip Lounibos. Ecology of invasive mosquitoes: effects on resident species and on human health. *Ecology Letters*, 8(5):558–574, 2005.

Alperen Burak Kantas, Halil I Cobuloglu, and İ Esra Büyüktahtakın. Multi-source capacitated lot-sizing for economically viable and clean biofuel production. *Journal of Cleaner Production*, 94:116–129, 2015.

Hans Kellerer, Ulrich Pferschy, and David Pisinger. Multidimensional knapsack problems. In *Knapsack Problems*, pages 235–283. Springer, Berlin, DE, 2004.

Cliff C. Kerr, Robyn M. Stuart, Dina Mistry, Romesh G. Abeysuriya, Gregory Hart, Katherine Rosenfeld, Prashanth Selvaraj, Rafael C. Nunez, Brittany Hagedorn, Lauren George, Amanda Izzo, Anna Palmer, Dominic Delport, Carrie Bennette, Bradley Wagner, Stewart Chang, Jamie A. Cohen, Jasmina Panovska-Griffiths, Michal Jastrzkebski, Assaf P. Oron, Edward Wenger, Michael Famulare, and Daniel J. Klein. Covasim: an agent-based model of covid-19 dynamics and interventions. *medRxiv*, 2020. doi: 10.1101/2020.05.10.20097469. URL https://www.medrxiv.org/content/early/2020/05/15/2020.05.10.20097469.

Eyyüb Y Kıbış and İ Esra Büyüktahtakın. Optimizing invasive species management: A mixed-integer linear programming approach. *European Journal of Operational Research*, 259(1):308–321, 2017.

Eyyüb Y Kıbış and İ Esra Büyüktahtakın. Optimizing multi-modal cancer treatment under 3D spatio-temporal tumor growth. *Mathematical Biosciences*, 307:53–69, 2019.

Eyyüb Y. Kıbış, İ Esra Büyüktahtakın, Robert G. Haight, Najmaddin Akhundov, Kathleen Knight, and Charlie Flower. A multi-stage stochastic programming approach to the optimal surveillance and control of emerald ash borer in cities. *Institute for Operations Research and the Management Science Journal on Computing*, 33(2):808–834, 2021.

Le-Minh Kieu, Nicolas Malleson, and Alison Heppenstall. Dealing with uncertainty in agent-based models for short-term predictions. *Royal Society Open Science*, 7(1):191074, 2020.

Kathleen S Knight, Daniel Herms, Reid Plumb, Eileen Sawyer, Daniel Spalink, Elizabeth Pisarczyk, Bernadette Wiggin, Rachel Kappler, Emily Ziegler, and Karen Menard. Dynamics of surviving ash (fraxinus spp.) populations in areas long infested by emerald ash borer (agrilus planipennis). In *Proceedings of the fourth international workshop on the genetics of host-parasite interactions in forestry: Disease and insect resistance in forest trees.*, volume 240, pages 143–152, 2012.

Frank H Koch, Denys Yemshanov, Manuel Colunga-Garcia, Roger D Magarey, and William D Smith. Potential establishment of alien-invasive forest insect species in the united states: where and how many? *Biological Invasions 13:969-985*, 13:969–985, 2011.

Walter D Koenig, Andrew M Liebhold, David N Bonter, Wesley M Hochachka, and Janis L Dickinson. Effects of the emerald ash borer invasion on four species of birds. *Biological Invasions*, 15(9):2095–2103, Sep 2013.

Varun Kompella, Roberto Capobianco, Stacy Jong, Jonathan Browne, Spencer Fox, Lauren Meyers, Peter Wurman, and Peter Stone. Reinforcement learning for optimization of covid-19 mitigation policies. *arXiv preprint arXiv:2010.10560*, 2020.

Weiwei Kong, Christopher Liaw, Aranyak Mehta, and D Sivakumar. A new dog learns old tricks: Rl finds classic optimization algorithms. In *International Conference on Learning Representations*, 2018.

Weiwei Kong, Christopher Liaw, Aranyak Mehta, and D Sivakumar. A new dog learns old tricks: Rl finds classic optimization algorithms. In *International Conference on Learning Representations*, Vancouver, CA, 2019.

Wouter Kool, Herke Van Hoof, and Max Welling. Attention, learn to solve routing problems! *arXiv preprint arXiv:1803.08475*, 2018.

Kent Kovacs, Robert Haight, Deborah Mccullough, Rodrigo Mercader, Nathan Siegert, and Andrew Liebhold. Cost of potential emerald ash borer damage in us communities, 2009-2019. *Ecological Economics*, 69:569–578, 01 2010. doi: 10.1016/j.ecolecon.2009.09.004.

Kent F Kovacs, Robert G Haight, Rodrigo J Mercader, and Deborah G McCullough. A bioeconomic analysis of an emerald ash borer invasion of an urban forest with multiple jurisdictions. *Resource and Energy Economics*, 36(1):270–289, 2014.

Jae Sik Lee and Monique Guignard. Note—an approximate algorithm for multidimensional zero-one knapsack problems—a parametric approach. *Management Science*, 34(3):402–410, 1988.

Fengcun Li and Bo Hu. Deepjs: Job scheduling based on deep reinforcement learning in cloud data center. In *Proceedings of the 2019 4th International Conference on Big Data and Computing*, pages 48–53, Honolulu, HI, 2019.

Junjiang Li, Philippe Giabbanelli, et al. Returning to a normal life via covid-19 vaccines in the united states: A large-scale agent-based simulation study. *JMIR Medical Informatics*, 9(4):e27419, 2021.

Xiaoshu Li, Thomas P Holmes, Kevin J Boyle, Ellen V Crocker, and C Dana Nelson. Hedonic analysis of forest pest invasion: the case of emerald ash borer. *Forests*, 10(9):820, 2019.

Haiguang Liao, Wentai Zhang, Xuliang Dong, Barnabas Poczos, Kenji Shimada, and Levent Burak Kara. A deep reinforcement learning approach for global routing. *Journal of Mechanical Design*, 142(6), 2020.

James H. Lorie and Leonard J. Savage. Three Problems in Rationing Capital. *The Journal of Business*, 28:229–229, 1955. doi: 10.1086/294081. URL `https://ideas.repec.org/a/ucp/jnlbus/v28y1955p229.html`.

Richard Loulou and Eleftherios Michaelides. New greedy-like heuristics for the multidimensional 0-1 knapsack problem. *Operations Research*, 27(6):1101–1114, 1979.

Guglielmo Lulli and Suvrajeet Sen. A branch-and-price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. *Management Science*, 50(6):786–796, 2004.

Qiang Ma, Suwen Ge, Danyang He, Darshan Thaker, and Iddo Drori. Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning, 2019.

Charles M Macal and Michael J North. Tutorial on agent-based modeling and simulation. In *Proceedings of the Winter Simulation Conference, 2005.*, pages 14–pp. Institute of Electrical and Electronics Engineers, 2005.

Charles M Macal and Michael J North. Agent-based modeling and simulation. In *Proceedings of the 2009 Winter Simulation Conference (WSC)*, pages 86–98. Institute of Electrical and Electronics Engineers, 2009.

Michael J Magazine and Osman Oguz. A heuristic algorithm for the multidimensional zero-one knapsack problem. *European Journal of Operational Research*, 16(3): 319–326, 1984. ISSN 0377-2217. doi: https://doi.org/10.1016/0377-2217(84)90286-8. URL `https://www.sciencedirect.com/science/article/pii/0377221784902868`.

Mufti Mahmud, Mohammed Shamim Kaiser, Amir Hussain, and Stefano Vassanelli. Applications of deep learning and reinforcement learning to biological data. *Institute of Electrical and Electronics Engineers Transactions on Neural Networks and Learning Systems*, 29(6):2063–2079, 2018.

Renata Mansini and M Grazia Speranza. Coral: An exact algorithm for the multidimensional knapsack problem. *Institute for Operations Research and the Management Science Journal on Computing*, 24(3):399–415, 2012. doi: 10.1287/ijoc.1110.0460. URL `https://doi.org/10.1287/ijoc.1110.0460`.

Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

Deborah G McCullough and Steven A Katovich. Pest alert: emerald ash borer. *United States Forest Services*, 2004.

Deborah G McCullough, Therese M Poland, Andrea C Anulewicz, and David Cappaert. Emerald ash borer (coleoptera: Buprestidae) attraction to stressed or baited ash trees. *Environmental Entomology*, 38(6):1668–1679, 2009.

Vicky McKeever. The coronavirus is expected to have cost 400 million jobs in the second quarter, un labor agency estimates. `https://www.cnbc.com/2020/06/30/coronavirus-expected-to-cost-400-million-jobs-in-the-second-quarter.html`, 2020. Accessed: 07-06-2021.

Daniel W McKenney, John H Pedlar, Denys Yemshanov, D Barry Lyons, Kathy L Campbell, and Kevin Lawrence. Estimates of the potential cost of emerald ash borer (agrilus planipennis fairmaire) in canadian municipalities. *Arboriculture and Urban Forestry*, 38(3):81, 2012a.

Daniel W McKenney, John H Pedlar, Denys Yemshanov, Donald Barry Lyons, Kathy L Campbell, and Kevin Lawrence. Estimates of the potential cost of emerald ash borer (agrilus planipennis fairmaire) in canadian municipalities. 2012b.

Sanjay Mehrotra, Hamed Rahimian, Masoud Barah, Fengqiao Luo, and Karolina Schantz. A model of supply-chain decisions for resource sharing with an application to ventilator allocation to combat covid-19. *Naval Research Logistics*, 67(5):303–320, 2020.

Shefali V Mehta, Robert G Haight, Frances R Homans, Stephen Polasky, and Robert C Venette. Optimal detection and control strategies for invasive species management. *Ecological Economics*, 61(2-3):237–245, 2007.

Rodrigo J Mercader, Deborah G McCullough, Andrew J Storer, John M Bedford, Robert Heyd, Therese M Poland, and Steven Katovich. Evaluation of the potential use of a systemic insecticide and girdled trees in area wide management of the emerald ash borer. *Forest Ecology and Management*, 350: 70–80, 2015.

Rodrigo J Mercader, Deborah G McCullough, Andrew J Storer, John M Bedford, Robert Heyd, Nathan W Siegert, Steven Katovich, and Therese M Poland. Estimating local spread of recently established emerald ash borer, agrilus planipennis, infestations and the potential to influence it with a systemic insecticide and girdled ash trees. *Forest Ecology and Management*, 366:87–97, 2016.

Naomi Miller and Andrzej Ruszczyński. Risk-averse two-stage stochastic linear programming: Modeling and decomposition. *Operations Research*, 59(1): 125–132, 2011.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937. PMLR, 2016.

Seyed M Moghadas, Thomas N Vilches, Kevin Zhang, Shokoofeh Nourbakhsh, Pratha Sah, Meagan C Fitzpatrick, and Alison P Galvani. Evaluation of covid-19 vaccination strategies with a delayed second dose. *PLoS Biology*, 19(4): e3001211, 2021.

Randall S Morin and Andrew M Liebhold. Invasions by two non-native insects alter regional forest species composition and successional trajectories. *Forest Ecology and Management*, 341:67–74, 2015.

Sebastian A. Müller, Michael Balmer, William Charlton, Ricardo Ewert, Andreas Neumann, Christian Rakow, Tilmann Schlenther, and Kai Nagel. Predicting the effects of covid-19 related interventions in urban settings by combining activity-based modelling, agent-based simulation, and mobile phone data. *medRxiv*, 2021. doi: 10.1101/2021.02.27.21252583. URL `https://www.medrxiv.org/content/early/2021/03/01/2021.02.27.21252583`.

Navonil Mustafee, Sally Brailsford, Anatoli Djanatliev, Tillal Eldabi, Martin Kunc, and Andreas Tolk. Purpose and benefits of hybrid simulation: contributing to the convergence of its definition. In *2017 Winter Simulation Conference (WSC)*, pages 1631–1645. Institute of Electrical and Electronics Engineers, 2017.

Mohammadreza Nazari, Afshin Oroojlooy, Lawrence Snyder, and Martin Takác. Reinforcement learning for solving the vehicle routing problem. In *Advances in Neural Information Processing Systems*, pages 9839–9849, Montreal, CA, 2018.

Konstantinos Nikolopoulos, Sushil Punia, Andreas Schäfers, Christos Tsinopoulos, and Chrysovalantis Vasilakis. Forecasting and planning during a pandemic: Covid-19 growth rates, supply chain disruptions, and governmental decisions. *European Journal of Operational Research*, 290(1):99–115, 2021.

NJ. Covid-19 information hub. `https://covid19.nj.gov/forms/datadashboard`, 2021. Accessed: 07-06-2021.

Hazem AA Nomer, Khalid Abdulaziz Alnowibet, Ashraf Elsayed, and Ali Wagdy Mohamed. Neural knapsack: A neural network based solver for the knapsack problem. *Institute of Electrical and Electronics Engineers Access*, 8:224200–224210, 2020.

Matthias P Nowak and Werner Römisch. Stochastic lagrangian relaxation applied to power scheduling in a hydro-thermal system under uncertainty. *Annals of Operations Research*, 100(1-4):251–272, 2000.

Włodzimierz Ogryczak and Andrzej Ruszczyński. From stochastic dominance to mean-risk models: Semideviations as risk measures. *European Journal of Operational Research*, 116(1):33–50, 1999.

Włodzimierz Ogryczak and Andrzej Ruszczyński. On consistency of stochastic dominance and mean–semideviation models. *Mathematical Programming*, 89 (2):217–232, 2001.

Abu Quwsar Ohi, MF Mridha, Muhammad Mostafa Monowar, and Md Abdul Hamid. Exploring optimal control of epidemic spread using reinforcement learning. *Scientific Reports*, 10(1):1–19, 2020.

Sevilay Onal, Najmaddin Akhundov, İ Esra Büyüktahtakın, Jennifer Smith, and Gregory R Houseman. An integrated simulation-optimization framework to optimize search and treatment path for controlling a biological invader. *International Journal of Production Economics*, 222:107507, 2020.

Sevilay Onal, Sabah Bushaj, Esra Buyuktahtakin, and Gregory Houseman. A gaussian dispersal approach to capture long-term and long-distance dispersal through simulation-optimization. *To be submitted*, 222:107507, 2021. ISSN 0925-5273.

Afshin Oroojlooyjadid, MohammadReza Nazari, Lawrence Snyder, and Martin Takác. A deep q-network for the beer game: A reinforcement learning algorithm to solve inventory optimization problems. *arXiv preprint arXiv:1708.05924*, 2017.

Afshin Oroojlooyjadid, Lawrence V Snyder, and Martin Takáč. Applying deep learning to the newsvendor problem. *Institute of Industrial and Systems Engineers Transactions*, 52(4):444–463, 2020.

Jon Osthus. Tracking the eab infestation core 2017. results from the ash health, eab and eab bioagent monitoring study (300 tree study), 2017, 2017.

Bernardo K Pagnoncelli and Adriana Piazza. The optimal harvesting problem under price uncertainty: the risk averse case. *Annals of Operations Research*, 258(2): 479–502, 2017.

Dean R Paini, Andy W Sheppard, David C Cook, Paul J De Barro, Susan P Worner, and Matthew B Thomas. Global threat to agriculture from invasive species. *Proceedings of the National Academy of Sciences*, 113(27):7575–7579, 2016.

Liba Pejchar and Harold A Mooney. Invasive species, ecosystem services and human well-being. *Trends in Ecology and Evolution*, 24(9):497–504, 2009.

Georg Ch Pflug and Alois Pichler. Time-inconsistent multistage stochastic programs: Martingale bounds. *European Journal of Operational Research*, 249(1):155–163, 2016.

Andrew B Philpott and Vitor L De Matos. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. *European Journal of Operational Research*, 218(2):470–483, 2012.

David Pimentel, Rodolfo Zuniga, and Doug Morrison. Update on the environmental and economic costs associated with alien-invasive species in the united states. *Ecological Economics*, 52(3):273–288, 2005.

Hasan Pirkul. A heuristic solution procedure for the multiconstraint zero-one knapsack problem. *Naval Research Logistics (NRL)*, 34(2):161–172.

Hasan Pirkul. A heuristic solution procedure for the multiconstraint zero-one knapsack problem. *Naval Research Logistics (NRL)*, 34(2):161–172, 1987.

David Pisinger. A minimal algorithm for the 0-1 knapsack problem. *Operations Research*, 45(5):758–767, 1997.

Therese M Poland and Deborah G McCullough. Emerald ash borer: invasion of the urban forest and the threat to north america's ash resource. *Journal of Forestry*, 104(3):118–124, 2006.

Pierpaolo Pontrandolfo, Abhijit Gosavi, O Geoffrey Okogbaa, and Tapas K Das. Global supply chain management: a reinforcement learning approach. *International Journal of Production Research*, 40(6):1299–1317, 2002.

Padam Bahadur Poudel, Mukti Ram Poudel, Aasish Gautam, Samiksha Phuyal, Chiran Krishna Tiwari, Nisha Bashyal, and Shila Bashyal. Covid-19 and its global impact on food and agriculture. *Journal of Biology and Today's World*, 9(5):221–225, 2020.

Maciel M Queiroz, Dmitry Ivanov, Alexandre Dolgui, and Samuel Fosso Wamba. Impacts of epidemic outbreaks on supply chains: mapping a research agenda amid the covid-19 pandemic through a structured literature review. *Annals of Operations Research*, pages 1–38, 2020.

ZI Quick, GR Houseman, and İ Esra Büyüktahtakin. Assessing wind and mammals as seed dispersal vectors in an invasive legume. *Weed Research*, 57(1):35–43, 2017.

Roberto Rocha. What countries did right and wrong in responding to the pandemic. `https://www.cbc.ca/news/canada/covid-19-coronavirus-pandemic-countries-response-1.5617898`, 2020. Accessed: 07-06-2021.

Tyrrell Rockafellar and Stanislav Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2:21–42, 2000.

Tyrrell Rockafellar and Roger Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research*, 16(1):119–147, 1991.

Andrzej Ruszczyński and Alexander Shapiro. Conditional risk mappings. *Mathematics of Operations Research*, 31(3):544–561, 2006.

Krista Ryall. Detection and sampling of emerald ash borer (Coleoptera: Buprestidae) infestations. *The Canadian Entomologist*, 147(3):290–299, 2015.

Krista L Ryall, Jeffrey G Fidgen, and Jean J Turgeon. Detectability of the emerald ash borer (coleoptera: Buprestidae) in asymptomatic urban trees by using branch samples. *Environmental Entomology*, 40(3):679–688, 2011.

Krista L Ryall, Jeffrey G Fidgen, Peter J Silk, and Taylor A Scarr. Efficacy of the pheromone (3z)-lactone and the host kairomone (3z)-hexenol at detecting early infestation of the emerald ash borer, a grilus planipennis. *Entomologia Experimentalis et Applicata*, 147(2):126–131, 2013.

Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Rüdiger Schultz and Stephan Tiedemann. Conditional value-at-risk in stochastic programs with mixed-integer recourse. *Mathematical Programming*, 105(2-3): 365–386, 2006.

Suvrajeet Sen. Algorithms for stochastic mixed integer programming models. *Handbooks in Operations Research and Management Science*, 12:515–558, 2005.

Shizuo Senju and Yoshiaki Toyoda. An approach to linear programming with 0-1 variables. *Management Science*, pages B196–B207, 1968.

Md Salman Shamil, Farhanaz Farheen, Nabil Ibtehaz, Irtesam Mahmud Khan, and M Sohel Rahman. An agent-based modeling of covid-19: validation, analysis, and recommendations. *Cognitive Computation*, pages 1–12, 2021.

Alexander Shapiro. Time consistency of dynamic risk measures. *Operations Research Letters*, 40(6):436–439, 2012.

Alexander Shapiro, Wajdi Tekaya, Joari Paulo da Costa, and Murilo Pereira Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224(2):375–391, 2013.

Wei Shih. A branch and bound method for the multiconstraint zero-one knapsack problem. *Journal of the Operational Research Society*, 30(4):369–378, 1979.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Hamed Soleimani and Kannan Govindan. Reverse logistics network design and planning utilizing conditional value at risk. *European Journal of Operational Research*, 237(2):487–497, 2014.

Stephanie Soucheray. Us job losses due to covid-19 highest since great depression. `https://www.cidrap.umn.edu/news-perspective/2020/05/us-job-losses-due-covid-19-highest-great-depression`, 2020. Accessed: 07-06-2021.

Yunhao Tang, Shipra Agrawal, and Yuri Faenza. Reinforcement learning for integer programming: Learning to cut, 2020.

Matthew W Tanner, Lisa Sattenspiel, and Lewis Ntaimo. Finding optimal vaccination strategies under parameter uncertainty using stochastic programming. *Mathematical Biosciences*, 215(2):144–151, 2008.

Md Sarower Tareq, Tanzilur Rahman, Mokarram Hossain, and Peter Dorrington. Additive manufacturing and the covid-19 challenges: An in-depth study. *Journal of Manufacturing Systems*, 2021.

Reis Thebault, Tim Meko, and Junne Alcantara. Sorrow and stamina, defiance and despair. it's been a year. `https://www.washingtonpost.com/nation/interactive/2021/coronavirus-timeline/`, 2021. Accessed: 07-06-2021.

Arne Thesen. Scheduling of computer programs for optimal machine utilization. *BIT Numerical Mathematics*, 13:206–216, 1973.

Arne Thesen. A recursive branch and bound algorithm for the multidimensional knapsack problem. *Naval Research Logistics Quarterly*, 22(2):341–353, 1975.

Philia Tounta. Pandemic 2020: The impact on tourism and the shadowy points. `https://www.traveldailynews.com/post/pandemic-2020-the-impact-on-tourism-and-the-shadowy-points`, 2020. Accessed: 07-06-2021.

Jean J Turgeon, Jeffrey G Fidgen, Krista L Ryall, and Taylor A Scarr. Estimates of emerald ash borer (coleoptera: Buprestidae) larval galleries in branch samples from asymptomatic urban ash trees (oleaceae). *The Canadian Entomologist*, 148(3):361–370, 2016.

Johns Hopkins University. COVID-19 dashboard by the center for systems science and engineering (csse). `https://coronavirus.jhu.edu/map.html`, 2021. Accessed: 07-06-2021.

USDA. Confirmed emerald ash borer (EAB) activity in new jersey, 2018. URL `https://www.nj.com/news/2017/07/24_million_nj_trees_might_be_killed_this_year_and_we_cant_stop_it.html`. Accessed April 8th, 2020.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Michel Vasquez and Yannick Vimont. Improved results on the 0–1 multidimensional knapsack problem. *European Journal of Operational Research*, 165(1):70–81, 2005. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2004.01.024. URL `https://www.sciencedirect.com/science/article/pii/S0377221704000608`.

Michel Vasquez, Jin-Kao Hao, et al. A hybrid approach for the 0-1 multidimensional knapsack problem. In *In Proceedings of the International Joint Conference on Artificial Intelligence 2001*, pages 328–333, 2001.

Vijay V Vazirani. *Approximation algorithms*. Springer Science and Business Media, Chester, UK, 2013.

Richa Verma, Aniruddha Singhal, Harshad Khadilkar, Ansuma Basumatary, Siddharth Nayak, Harsh Vardhan Singh, Swagat Kumar, and Rajesh Sinha. A generalized reinforcement learning algorithm for online 3d bin-packing. *arXiv preprint arXiv:2007.00463*, 2020.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *arXiv preprint arXiv:1506.03134*, 2015.

Runzhe Wan, Xinyu Zhang, and Rui Song. Multi-objective reinforcement learning for infectious disease control with application to covid-19 spread. *arXiv preprint arXiv:2009.04607*, 2020.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3-4): 279–292, 1992.

H Martin Weingartner. Capital budgeting of interrelated projects: survey and synthesis. *Management Science*, 12(7):485–516, 1966.

H Martin Weingartner and David N Ness. Methods for the solution of the multidimensional 0/1 knapsack problem. *Operations Research*, 15(1):83–103, 1967.

Roger J-B Wets. Stochastic programs with fixed recourse: The equivalent deterministic program. *Society for Industrial and Applied Mathematics Review*, 16 (3):309–339, 1974.

David S Wilcove, David Rothstein, Jason Dubow, Ali Phillips, and Elizabeth Losos. Quantifying threats to imperiled species in the united states. *BioScience*, 48 (8):607–615, 1998.

Barry Tyler Wilson, Andrew J Lister, Rachel I Riemann, and Douglas M Griffith. Live tree species basal area of the contiguous united states (2000-2009). 2013. doi: https://doi.org/10.2737/RDS-2013-0013.

Gerhard J Woeginger. Exact algorithms for np-hard problems: A survey. In *Combinatorial optimization—eureka, you shrink!*, pages 185–207. Springer, Berlin, DE, 2003.

Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *Advances in Neural Information Processing Systems*, 30:5279–5288, 2017.

Yongquan Zhou Yan Yang, Shengjian Liu. Greedy binary lion swarm optimization algorithm for solving multidimensional knapsack problem. *Journal of Computer Applications*, 40(5):1291–1294, 2020.

Ming-Hsien Yang. An efficient algorithm to allocate shelf space. *European Journal of Operational Research*, 131(1):107–118, 2001.

Yuwen Yang and Jayant Rajgopal. Learning combined set covering and traveling salesman problem, 2020.

Denys Yemshanov, Frank H Koch, Mark Ducey, and R Venette. Making invasion models useful for decision makers: incorporating uncertainty, knowledge gaps, and decisionmaking preferences. *Pest Risk Modelling and Mapping for Invasive Alien Species*, 7:206–222, 2015.

Denys Yemshanov, Robert G Haight, Frank H Koch, Bo Lu, Robert Venette, Ronald E Fournier, and Jean J Turgeon. Robust surveillance and control of invasive species using a scenario optimization approach. *Ecological Economics*, 133: 86–98, 2017.

Denys Yemshanov, Robert G Haight, Frank H Koch, Robert C Venette, Tom Swystun, Ronald E Fournier, Mireille Marcotte, Yongguang Chen, and Jean J Turgeon. Optimizing surveillance strategies for early detection of invasive alien species. *Ecological Economics*, 162:87–99, 2019a.

Denys Yemshanov, Robert G Haight, Ning Liu, Cuicui Chen, Chris JK MacQuarrie, Krista Ryall, Robert Venette, and Frank H Koch. Acceptance sampling for cost-effective surveillance of emerald ash borer in urban environments. *Forestry: An International Journal of Forest Research*, 2019b.

Yin and İ Esra Büyüktahtakın. A multi-stage stochastic programming approach to epidemic resource allocation with equity considerations. *Under Review*, 2020.

Xuecheng Yin and İ Esra Büyüktahtakin. A multi-stage stochastic programming approach to epidemic resource allocation with equity considerations. *Health Care Management Science*, pages 1–26, 2021a.

Xuecheng Yin and İ Esra Büyüktahtakin. Risk-averse multi-stage stochastic programming to optimizing vaccine allocation and treatment logistics for effective epidemic response. *Institute of Industrial and Systems Engineers Transactions on Healthcare Systems Engineering*, (just-accepted):1–52, 2021b.

Xuecheng Yin, I Esra Buyuktahtakin, and Bhumi P Patel. Covid-19: Optimal allocation of ventilator supply under uncertainty and risk. *Available at SSRN 3801183*, 2021.

Weini Zhang, Hamed Rahimian, and Güzin Bayraksan. Decomposition algorithms for risk-averse multistage stochastic programs with application to water allocation under uncertainty. *Institute for Operations Research and the Management Science Journal on Computing*, 28(3):385–404, 2016.

Yu Zhang, Jianguo Yao, and Haibing Guan. Intelligent cloud resource management with deep reinforcement learning. *Institute of Electrical and Electronics Engineers Cloud Computing*, 4(6):60–69, 2017.