UPSCALING INFRASTRUCTURE ASSET MANAGEMENT SYSTEMS UNDER COMPLEX UNCERTAINTIES TO NETWORK ASSETS USING MACHINE LEARNING

VAHID ASGHARI

PhD

The Hong Kong Polytechnic University

2022

The Hong Kong Polytechnic University

Department of Civil and Environmental Engineering

Upscaling Infrastructure Asset Management Systems Under Complex Uncertainties to Network Assets Using Machine Learning

Vahid Asghari

A thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

November 2021

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

_____

Vahid Asghari

To

My family,

My beloved wife,

My mentors and teachers,

Abstract

Deteriorating and at-risk infrastructure assets should be maintained at acceptable conditions by infrastructure asset management (IAM) systems to ensure the safety and welfare of communities. Given the importance of maintaining infrastructure assets, asset management at both project- and network levels has been the focus of hundreds of studies. These IAM systems utilize probabilistic and non-linear models to accurately model various phenomena. With a commonly adopted framework using Monte Carlo simulation and heuristic optimization algorithms, IAM systems proposed in the literature aim to maintain the functionality of assets in their life cycle by optimally allocating limited resources to different intervention actions in time.

Although these studies and previously developed asset management systems have pushed the boundaries of this field, they are subject to a number of limitations. First, these advances have been made separately without the ability to build upon one another. Second, project-level IAM (PL-IAM) is capable of optimizing maintenance interventions in the life cycle of assets by incorporating probabilistic and complex models but at the expense of relatively high computation time. Due to their high computational costs, upscaling complex PL-IAM systems to a multitude of assets is currently far from practical. Consequently, uncertainties have usually been simplified in this problem to reach acceptable strategies in feasible computational time. This simplification could lead to sub-optimal and far from reality strategies. Also, the derived optimal plans lack flexibility in decision-making in face of unprecedented uncertainties. Ever-changing uncertain phenomena might render previously stochastic models obsolete and, therefore, optimized MRR plans sub-optimal. In a more recent line of research, reinforcement learning (RL) has been adopted by IAM researchers for adding flexibility regarding uncertainties in preventive actions decision-making. As the third

major limitation highlighted in this dissertation, this line of research lacks (1) incorporating other sources of uncertainties, such as hazards, apart from deterioration patterns, and (2) considering managerial aspects of IAM such as stakeholders' utilities. Fourth, the network-level infrastructure asset management (NL-IAM) problem, which is excessively large and complex due to a large number of decision parameters, possible strategies, and underlying uncertainties, has not gained proper attention in the RL-based IAM studies.

This dissertation puts forward the first open-source, extensible, freely accessible, and modular platform developed in Python, GIAMS, paving the way for researchers and practitioners to easily collaborate and contribute to future related research. Drawing upon related literature, it describes modules of GIAMS and illustrates their use with bridge components and models. Aiming to address the gap between the literature and practice of PL-IAM, which is making complex PL-IAM systems computationally applicable to all assets in a network, this dissertation presents a methodology to replace the time-consuming simulation modules of optimization algorithms with a trained machine learning model estimating life cycle costs analysis (LCCA) results. Then, a new machine learning-based methodology is presented to learn the behavior of optimization algorithms and estimate (near-)optimal intervention timings instead of doing the optimization. Next, this study provides a holistic framework, from the early development of IAM systems and microworlds to training RL models and verification of results. This framework focuses on considering deterioration, hazards, and costs fluctuations as the main sources of uncertainties and adopting managerial aspects into decision making. Finally, one of the first RL-based NL-IAM systems is put forward for achieving intervention strategies while considering different sources of uncertainties such as earthquake, deterioration, and costs fluctuations.

One project-level lifecycle optimization and one network-level project selection based on the Indiana, US, bridge network with more than 4,600 bridges are provided to demonstrate

the applicability of GIAMS. Then, deep neural network (DNN) models were trained on the LCCA results of more than 1.4 million semi-synthesized bridges based on the US NBI considering different intervention actions and uncertainties about condition ratings, hazards, and costs. The findings show that the trained DNN models can accurately estimate the complex LCCA results 5 orders of magnitudes faster than simulation techniques. As the next step, an ensemble of random forests models was trained on optimal maintenance timings of more than 1.6 million semi-synthesized bridges to illustrate the proposed methodology for directly estimating optimal plans. The trained model could yield optimized MRR plans, with more than 95% accuracy on the test set and more than 89% accuracy on the real highway bridges of Indiana with more than 4600 assets, 6 orders of magnitudes faster than the conventional framework of complex MRR optimization. Given the proposed RL-based framework for IAM, multi-agents RL models, based on DQN and actor-critic models, are constructed and trained for taking intervention actions regarding elements of a real bridge in Indiana, the US, through its life cycle. Both models could increase the utilities by up to 14% and decrease the costs in the project-level analysis. Alongside the flexibility in decision-making provided by the trained A2C models for NL-IAM, the results show that more favorable strategies in terms of stakeholders' utilities could be achieved using the proposed framework. Effective solutions for tackling the curse of dimensionality and reward engineering are also provided herein.

The modular and open-source nature of the presented software (GIAMS) makes it readily capable of further extension in a variety of asset management topics. The proposed methodology for estimating LCCA results help practitioners reduce the optimization and LCCA computation times of complex IAM systems to a feasible level for practical utilization. Practitioners can adopt the proposed methodology for predicting optimal plans to enhance their decision-making systems, obtain optimal maintenance plans without sacrificing

complex and accurate models, and take another step towards sustainability objectives. Consistent with the existing practice of IAM, the proposed RL-based framework brings flexibility in face of uncertainties to the IAM decision-making process. In other words, the trained MARL models can assist decision-makers with making decisions that are close to reality while considering complex models in a short time in the IAM domains. The proposed framework and solutions for RL-based NL-IAM can pave the path for researchers toward enhancing the current state of NL-IAM frameworks. Trained multi-agent A2C models can inform asset managers with potentially more profitable, more desirable, and closer to reality strategies leading to community-level enhancements in sustainability measures.

List of papers

The dissertation is based on the following papers:

I. **Asghari, V.** and Hsu, S.-C. (2022) 'Network-level infrastructure asset management with multi-agent actor-critic reinforcement learning', Manuscript

II. **Asghari, V.,** Jahan Biglari, A., Hsu, S.-C., (2022) 'Multi-agent reinforcement learning for project-level intervention planning under multiple uncertainties', Automation in Construction, Under review paper

III. **Asghari, V.,** Wang, Y., Jahan Biglari, A., Hsu, S.-C., Tang, P. (2022) 'Reinforcement learning in construction engineering and management: Opportunities and challenges', Journal of Construction Engineering and Management, Under review paper

IV. **Asghari, V.,** and Hsu, S.-C. (2022) 'Upscaling complex project-level infrastructure intervention planning to network assets', Journal of Construction Engineering and Management, 18(1), doi: 10.1061/(ASCE)CO.1943-7862.0002221

V. **Asghari, V.,** Hsu, S.-C. and Wei, H.-H. (2021) 'Expediting life cycle cost analysis of infrastructure assets under multiple uncertainties by deep neural networks', Journal of Management in Engineering, 37(6). doi: 10.1061/(ASCE)ME.1943-5479.0000950.

VI. **Asghari, V.** and Hsu, S.-C. (2021) 'An open-source and extensible platform for general infrastructure asset management system', Automation in Construction, 127. doi: 10.1016/j.autcon.2021.103692.

Other journal papers of the author during the Ph.D. program not included in this thesis:

VII. Wang, R., **Asghari, V.,** Cheung, C. M., Hsu, S.-C, Lee, C.-J., (2021) 'Assessing Effects of Economic Factors on Construction Cost Estimation using Deep Neural Networks', Automation in Construction, 134, doi: 10.1016/j.autcon.2021.104080

VIII. **Asghari, V.,** Kashani, H. and Hsu, S.-C. (2021) 'Optimal timing of the seismic vulnerability reduction measures using real options analysis', ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering. doi: 10.1061/AJRUA6.0001146.

IX. Wang, R., **Asghari, V.,** Hsu, S.-C., Lee, C.-J, Chen, J,-H, (2020) 'Detecting corporate misconduct through random forest in China's construction industry', Journal of Cleaner Production. Elsevier Ltd, 268, p. 122266. doi: 10.1016/j.jclepro.2020.122266.

X. **Asghari, V.,** Y.F. Leung, S.-C. Hsu (2020) 'Deep neural network based framework for complex correlations in engineering metrics', Advanced Engineering Informatics. 44 101058. doi:10.1016/j.aei.2020.101058.

Conference proceedings:

XI. Alvarado, V., Hsu, S.-C., **Asghari, V.,** Huang, X., Lee, P., "Autotrophic Nitrogen Removal Prediction by Machine Learning in a Partial Nitrification/Anammox Fluidized-bed Membrane Bioreactor", 1st Asia Pacific Conference on Sustainable Development of Energy, Water and Environment Systems, Australia, 2020

XII. **Asghari, V.,** Leung, A., S.C. Hsu, "Deep Neural Networks for Prediction of Undrained Shear Strength of Clays", 7th International Symposium on Geotechnical Safety and Risk, Taiwan, Dec2019

Acknowledgments

I would like to offer my most sincere and deepest appreciation to my supervisor, mentor, and teacher Professor Shu-Chien Hsu and his perpetual support during my studies. His guidance, thoughtful discussions, experience, mentorship, and academic excellence have improved both my personality and my research immensely. Working under his supervision has been, and will ever be, a great honor.

I would also like to express my gratitude to my co-supervisor, Professor Anthony Chen, and my confirmation examination committee members, Professor Zhen Leng and Professor Tony Sze, for their insightful, helpful, and constructive comments and criticisms. Also, I would like to sincerely appreciate the expert and important comments received from Professor William Lam. A considerable portion of the quality of this thesis is indebted to their expert opinions and comments. My Ph.D. journey wouldn't have been accomplished without the financial support of The Hong Kong Polytechnic University and the general support of the staff working in the faculty of Construction and Environment.

I hereby intend to acknowledge the life-long support of my family leading to my current state. My deepest appreciation to my wife Fatemeh who supported me unconditionally during these years. I also want to offer my appreciation to our research group members Ran Wang, Valeria Alvarado, Zhongnan Ye, Ziyue Yuan, Can Chen, and Yunping Huang whose support and friendship made this path easier.

It is impossible to properly acknowledge all who had a share in helping me finish this research and dissertation. Among all, it is necessary to acknowledge the role of my lifelong mentors and teachers Dr. Hammed Kashani, Dr. Mojtaba Mahsuli, Dr. Morteza Eskandari, and Dr. Omid Ahanchi. Also, special thanks to all my friends, especially the close ones, who accompanied me my whole life.

Table of contents

List of tables

List of figures

List of abbreviations

Abbreviation	Definition
A2C	Advantage Actor-Critic
A3C	Asynchronous Advantage Actor-Critic
ADP	Approximate Dynamic Programming
ADT	Average Daily Traffic
AI	Artificial Intelligence
BDLM	Bayesian Dynamic Linear Model
BMS	Bridge Management System
CART	Classification And Regression Trees
CEM	Construction Engineering and Management
CR	Condition Rating
DCMAC	Deep Centralized Multi-Agent Actor-Critic
DDPG	Deep Deterministic Policy Gradient
DMA2C	Decentralized Multi-Agent Advantage Actor-Critic
DNN	Deep Neural Networks
DQN	Deep Q-Networks
DT	Decision tree
FEMA	Federal Emergency Management Agency
FHWA	Federal Highway Administration
GA	Genetic Algorithm
GIAMS	General Infrastructure Asset Management System
GPU	Graphical Processor Units
HOA	Heuristic Optimization Methods
HVAC	Building Heat, Ventilation, and Air Conditioning
IAM	Infrastructure Asset Management
IBC	Incremental Benefit-Cost Ratio
IBMS	Indiana Bridge Management System
IMOMC	Imbalanced Multi-Output Multi-Class
IUC	Incremental Utility Costs
Lat.	Latitude
LCA	Life Cycle Assessment
LCC	Life Cycle Costs
LCCA	Life Cycle Cost Analysis
Long.	Longitude
MAE	Mean of Absolute Errors
MAPE	Mean of Absolute Percentage Error
MARL	Multi-Agent Reinforcement Learning
MCMDKP	Multichoice Multidimensional Knapsack Problem
MCS	Monte Carlo Simulation
MCTS	Monte Carlo Tree Search
MDPs	Markov Decision Processes

ML	Machine Learning
MOO	Multi-Objective Optimization
MRR	Maintenance, Rehabilitation, or Reconstruction
MSE	Mean of Squared Error
NBI	National Bridge Inventory
Nibs	National Institute of Building Sciences
NL-IAM	Network-Level Infrastructure Asset Management
OLS	Ordinary Least Squares
PG	Policy Gradient
PL-IAM	Project-Level Infrastructure Asset Management
POMDPs	Partially Observable Mdps
PPO	Proximal Policy Optimization
PPP	Poisson Point Process
PSO	Particle Swarm Optimization
RF	Random forests
RL	Reinforcement Learning
ROA	Real Options Analysis
SARSA	State–Action–Reward–State–Action
SOO	Single Objective Optimization
SVM	Support vector machines
TexasDOT	Texas Department of Transportation
U/C	Utility/Cost
USGS	United States Geological Survey

List of notations and symbols

Symbol	Definition	Unit
$\mathbf{A}$	Action vector in RL	-
A^{π}	Advantage function in RL algorithm	-
A_d^{π}	Advantage values in RL algorithm	-
$\mathbf{A_p}$	Asset parameters	-
A_d	Deck age	years
A_r	Regression coefficients	-
A_{sp}	Superstructure age	years
A_{sb}	Substructure age	years
A_t	A selected course of actions at t in the RL algorithm	-
B_d	Budget	$
$\mathbf{b}^{\mathrm{i}}$	Bias vector of layer i in neural networks	-
$B_{i,j}$	The pair of binary classification models for predicting MRR action	-
$B'_{i,j}$	The pair of binary classification models for predicting MRR action	-
C	Costs of projects	$
C^{π}	Critic value in RL algorithm	-
C_A	Agency costs	$
$\overline{C_A}$	Expected agency costs	$
C_d	A part of user costs due to travel delay	$
C_{d_P}	Value of travel time for personal vehicles	$/hr
C_{d_T}	Values of travel time for trucks	$/hr
C_f	The costs due to excessive fuel consumption	$
C_{f_P}	Marginal costs of personal vehicles fuel burn	$/km
C_{f_T}	Marginal costs of trucks fuel burn	$/km
$C_{L\|IM}$	Loss costs and given a certain hazard with an intensity measure of IM	$
C_M	Direct maintenance costs	$
$C_{R\|IM}$	Recovery costs given a certain hazard with an intensity measure of IM	$
C_T	MRR user costs	$
C_U	User costs	$
$\overline{C_U}$	Expected user costs	$
CR_d	Deck condition	-
CR_{pt}	Condition rating of an element after project p is conducted at t	-
CR_{sb}	Substructure condition	-
CR_{sp}	Superstructure condition	-
d	Maximum depth of branching during DT development	-
$\mathbf{D}$	Holder for data points in the sampling process	-
ds	Damage state	-
D_t	Design	-
e	The difference between the previously perceived Q-value and new results	-
f^L	Lower bands of constraints	-
f^U	Upper bands of constraints	-

$f_m(x)$	Objective function	-
f^{max}	Maximum features during RF development	-
FN	False-negative predicted labels in classification	-
FP	False-positive predicted labels in classification	-
g^{π}	Policy gradient function	-
$\mathbf{G}$	Each generation in the GA algorithm	-
G	Gini impurity index	-
$\mathbf{H}$	A hazard scenario in a life cycle	-
HZ	HAZUS class	-
I	Information gain index by entropy	-
IM	The ground motion intensity measure	-
L2	Regularization type in neural networks	-
$\mathbf{I}^{\mathrm{i}}$	Vectorized results of layer i in the neural network	-
L_b	The length of the bridge or MRR projects	m
L_d	The length of detour (alternate road)	m
$\mathbf{M}$	MRR plans	-
$\bar{\mathbf{M}}$	Estimated MRR plans	-
$\hat{\mathbf{M}}$	Optimal MRR plan with the highest objective value function	-
$\mathbf{M_j}$	A subset of a portfolio of actions to be conducted	-
$\boldsymbol{M_{j'}}$	The remaining set of actions in a portfolio to be postponed to the next time step	-
M_a	Material	-
M_d	Deck material	-
$M_{i,j}$	The multiclass classification model for predicting MRR actions	-
m_{S_i}	The median value of ground motion intensity with damage state i	-
$\mathrm{N}(\mathbf{X})$	Normalization function	-
N_p	Number of projects	-
N_s	Number of simulations	-
N_{sp}	Number of spans	-
N_l	Maximum remaining observations in a leaf during DT development	-
N_r	Minimum number of samples in a root node to be split during DT development	-
N_T	Number of decision-making steps in the investment horizon	-
N_{tr}	Number of trees	-
NPV	The net present value function	-
P	Probability	-
$\mathbf{P}$	A portfolio of projects	-
p	A selected project	-
P_b	Penalty for exceeding the available budget	-
p_d	The percentage of drivers that would use the detour	-
P_f	Penalty for futile actions	-
p_i	Percentage of labels in leaves during DT development	-
P_{ij}	Transition probability from state i to state j	-
$\mathbf{P_{IUC}}$	The selected portfolio of projects based on IUC	-

P_M	Markovian interstate model	-
$P_{S\geq S_i\|IM}$	The probability of exceedance of the state of assets from S_i given IM	-
$P_{S\geq S_i\|IM,S_j}$	The probability of exceedance of the state of assets from S_i given IM and S_j	-
p_T	The truck percentage	-
Q^π	Q-value function in Q-learning algorithm	-
$\mathbf{Q_{IUC}}$	All projects that are sorted based on IUC	-
r	Discount rate	-
$\mathbf{R}$	Reward vector in RL algorithm	-
R	Reward value in optimization and reward function in RL algorithm	-
$\bar{R}$	Expected reward in RL algorithm	-
$\mathbf{R_{LCCA}}$	LCCA results	-
$\bar{\mathbf{R}}_{\mathbf{LCCA}}$	Average of LCCA results	-
$\mathbf{RC}$	Vectorized recovery actions to be conducted as after-hazard strategies	-
$\overline{\mathbf{R_t}}$	Expected LCCA results	-
RCL	Road class	-
$\mathbf{S}$	State vector in RL	-
S^*	Elements of the state vector in the RL algorithm	-
S	State of the asset – A modified form of condition ratings	-
$\mathbf{SC}$	A randomized sample of all incidents in the life cycle of assets	-
Sl	Soil type class	-
$\mathbf{S_{MCS}}$	Simulation parameters	-
t	Time (Decision-making step in the investment horizon)	year
$\boldsymbol{t_{ma}}$	Maintenance duration	days
$\boldsymbol{t_{rc}}$	Reconstruction duration	days
$\boldsymbol{t_{rh}}$	Rehabilitation duration	days
T	Investment horizon	years
T_p	Project duration	days
TN	True negative predicted labels in classification	-
TP	True positive predicted labels in classification	-
U	Utility of actions	-
u_{DC}	Utility of deck	-
u_{SB}	Utility of substructure	-
u_{SP}	Utility of superstructure	-
u_1	Utility of elements before conducting MRR actions	-
u_2	Utility of elements after conducting MRR actions	-
UC	Utility over cost ratio	-
V^π	Value function in RL algorithm	-
V_a	Average speed prior to construction	km/hr
V_b	Average speed during construction	km/hr
V_c	Vertical clearance	M
V_d	Average speed in the detour	km/hr
W_b	Width of bridges	m
$\mathbf{W}^i$	Vectorized nodes' weight in the neural networks	-

W_t	Wiener process value at t	-
X_i	The binary value indicating the presence of project i in a portfolio	-
y_i	Actual values in a collected dataset to be predicted	-
$\hat{y}_i$	Predicted values of y_i using a model	-
Z	Confidence interval factor	-
α	Learning rate	-
α_r	Regression coefficients	-
$\beta_i(S_t)$	The dispersion factor of damage state i at time t	-
β_M	Memory buffer	-
β_r	Regression coefficients	-
γ	Discount factor	-
Γ	The rank of individuals in each generation of the GA optimization	-
ϵ	Exploration probability in RL algorithm	-
ϵ_r	Error term in machine learning prediction	-
E	Acceptable error for the confidence interval calculation	-
ζ	The occurrence rate of hazards (earthquakes)	events/ year
η	The drift ratio (the trend) of costs	-
θ	Skew angle	degree
Θ	An RL model	-
Θ'	A target RL model	-
Θ_O	Trained machine learning models for predicting optimal MRR plans	-
Θ_L	Trained machine learning models for predicting LCCA results	-
λ_ϵ	Decay factor for epsilon-greedy algorithm	-
μ_e	Earthquake magnitude distribution: Lognormal mean	-
$\mu_{\tilde{x}}$	Mean of the normal distribution	-
v_t	Value of the Wiener process with drift at t	-
π	Policy function in RL	-
σ	Standard deviation	-
σ_0	An initial estimate of standard deviation	-
σ_e	Earthquake magnitude distribution: Lognormal standard deviation	-
Φ	The standard normal cumulative distribution function	-
ϕ_M	Simulated microworld in RL algorithm	-
ϕ	Activation function	-
ψ^π	The general form of the Advantage function in the RL algorithm	-

Chapter 1 Introduction

Infrastructure assets are pillars of modern societies all over the world. In fact, reliable, well-functioning, and durable community infrastructure assets, such as transportation networks, water supply systems, and energy distribution networks, are indisputable requirements for economic growth, sustainable development, and prosperity in the modern world (Biondini and Frangopol, 2016). The condition of these assets is constantly altered by aging, loading, environmental agents, or even natural and man-made hazards. The need for continuous, uninterrupted, and optimum operation of these deteriorating infrastructure systems, highlight the importance of developing advanced methodologies and decision-making tools for minimizing the whole life cycle costs (LCC) incurred by communities. Agencies, asset managers, and decision-makers typically use infrastructure asset management (IAM) systems to maintain their assets in acceptable conditions and invest significant monetary resources to maintain infrastructure assets at a safe and operational level. Asset management refers to the process of condition monitoring, collecting data, analyzing, and decision making about resource allocation to the maintenance of assets (Sinha *et al.*, 2009). IAM studies and frameworks have been continuously under development to inform stakeholders and decision-makers with appropriate and accurate intervention plans. The term "appropriate" in this domain has a broad range of meanings from reducing long-term environmental impacts and reducing LCC to enhancing the safety of assets in face of different sources of hazards. Regardless of the exact definition of the term "appropriate", IAM systems must answer the following questions: 1) which components need to be maintained, 2) when, and 3) to what extent the intervention should be carried out (Lounis and McAllister, 2016) (i.e., regular maintenance, rehabilitation, or reconstruction(MRR)). Informed and precise decisions about evaluative and predictive tasks can minimize the risk of failure, save a significant amount of

monetary and human resources, and improve safety in construction engineering and management (CEM) projects. This underscores the importance of improving asset management systems globally (Kaganova and Telgarsky, 2018; Zhong, Ng and Skitmore, 2019).

Asset management is a difficult task for stakeholders mainly because of insufficient high-quality data, stochastic deterioration, infrastructure vulnerabilities, unprecedented natural hazards, varying levels of asset response to future hazards, restoration challenges after damage, and fluctuating market conditions (Yang *et al.*, 2019). The presence of these sources of uncertainties renders obtaining optimal difficult and even sometimes impossible. Despite the complexity and non-linearity of this problem, researchers and practitioners have not abandoned solving it. To optimally allocate limited monetary and manpower resources to different intervention actions in the life cycle of assets, decision-making methodologies with complex models for considering different phenomena and optimization algorithms for finding (near-)optimal strategies have been put forward in hundreds of studies at both project and network-level in the past two decades (Sánchez-Silva *et al.*, 2016; Chen and Bai, 2019; Khalifa *et al.*, 2019). Project-level infrastructure asset management (PL-IAM) is the process of optimizing alternative actions such as maintenance, rehabilitation, and reconstruction (MRR) in the life cycle of an asset. Network-level project management refers to the process of selecting the best subset of projects from a pool of projects given a limited budget to maximize stakeholders' utilities. Apart from the research papers, commercial computer programs (Hawk and Small, 1998; Thompson *et al.*, 1998) have also been developed for the maintenance of infrastructure assets.

The abovementioned asset management systems are associated with different strengths and limitations. However, there are generally three main limitations shared by all these IAM systems. First and foremost, they are not extensible. This means one cannot easily update a

component of others' asset management systems for further research and analysis. As a result, whoever intends to conduct a study in the asset management field must develop IAM systems almost from scratch. Second, the programming languages of these IAM systems that are being used in different departments of transportation are not flexible enough. Consequently, it is too difficult, and sometimes impossible, to apply new tools and models, such as machine learning (ML) models, to them. Third, some of the IAM systems that are used by agencies are licensed and expensive. Therefore, practitioners, both in academia and industry, in some parts of the world do not have access to the currently used IAM systems. In addition, municipalities in underprivileged countries will never be able to use up-to-date IAM systems for their assets due to the above-described constraints. Given these limitations, there is a need for an open-source asset management platform built with widely used and flexible programming languages.

Studies proposing PL-IAM systems have usually employed complex models to consider various phenomena affecting the asset and use various optimization techniques to find the best set of actions in the life cycle of an asset. The literature analysis on MRR planning of infrastructure assets shows that a major portion of studies has used heuristic optimization algorithms (HOA) and Monte Carlo simulation (MCS) for finding near-optimal solutions when various sources of uncertainties and complex models underlie the problem (Miyamoto, Kawamura and Nakamura, 2000; Frangopol, 2011; Lagaros, Kepaptsoglou and Karlaftis, 2013; Saydam and Frangopol, 2015; Arif, Bayraktar and Chowdhury, 2015; Chen *et al.*, 2015; Renna, 2017; Ahmad, Amr and Mario, 2018; Ahmad, Mario and Amr, 2018; Kim and Frangopol, 2018; France-Mensah and O'Brien, 2018; Ghodoosi *et al.*, 2018; Montazeri and Touran, 2019; Yang, Frangopol and Teng, 2019; Li *et al.*, 2020; Cheng, Yang and Frangopol, 2020; Elhadidy, Elbeltagi and El-Badawy, 2020). This approach (MCS-HOA) typically has two computational loops: 1) the simulation loop to model the uncertainties and evaluate the

expected life cycle cost analysis (LCCA) results (Yang, Hsieh and Kung, 2012; Salimi, Mawlana and Hammad, 2014), and 2) HOA loop to find the best set of actions under different constraints to optimize one or several objectives (Soliman, Ahmed and Tarek, 2017; Yang, Frangopol and Teng, 2019).

Depending on the type of asset management, this flexible framework (i.e., simulating life cycle events and optimizing actions) has its upsides and downsides. On the one hand, the framework allows researchers to assess and investigate the efficiency of various types of modeling techniques such as deterioration models, hazard response models, and optimization algorithms in project-level management. On the other hand, it is currently far from practical for application to each asset at the network level due to the framework's relatively high computational time (Frangopol, 2011). The computation time of PL-IAM systems that carry out Monte Carlo simulations for life cycle analysis and optimization using heuristic algorithms is usually in the order of minutes (Yang, Hsieh and Kung, 2012; Kim and Frangopol, 2018). However, this seemingly short amount of time becomes problematic if it is applied to a large network of assets. For example, if the MRR optimization of a bridge in its life cycle takes 5 minutes, it will take approximately 6 months to perform a similar process for the Texas network of bridges with more than 50,000 highway bridges. Needless to say, the more complex models and uncertain phenomena are implemented in the optimization problem, the greater number of samplings and computational time is required. As a result, asset managers and decision-makers usually have to strike a balance between the complexity of models and the computation time of decision-making frameworks for application in real life. The trade-off between complexity and computation time is conducted mainly due to the necessity of evaluating a multitude of strategies within a practicable time (Kandil, El-Rayes and El-Anwar, 2010; Frangopol, 2011; Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011; Yang, Hsieh and Kung, 2012; David Y. Yang and Frangopol, 2020a). Simplified models are

usually used for network-level optimization of assets in real-life applications (e.g., Thompson *et al.*, 1998; Sinha *et al.*, 2009; Yang and Frangopol, 2020). The simplification process would lead to intervention planning schemes based on life cycle results far from reality. Overall, these highlight the need for methodologies to reach more accurate and reliable results without sacrificing complex models in a feasible time.

However, unprecedented events, such as a new deterioration pattern in the assets' elements or asset responses to hazards, can make a previously optimal plan sub-optimal. In other words, these intervention planning solutions, whether a global optimum or a near-optimal plan, are associated with a common limitation: inflexibility in decision making toward future uncertain events. The optimized MRR plans are usually derived assuming the stochastic models would model different phenomena perfectly or in a near-perfect fashion. But, environmental factors and other interruptions can change the behaviors of infrastructure systems and the corresponding models. For example, an unknown source of chemicals in a stream can accelerate the deterioration of a bridge's submerged columns. Or, Changing the transportation routes such as constructing a new highway can significantly reduce the user costs of a previously commonly used bridge. The optimized MRR plans offer stakeholders rigid plans based on several assumptions that might not hold during the life cycle of assets. Several attempts have been made in the IAM domain to add flexibility to the derived optimal plans. Real options analysis (ROA) is an example of such attempts (Ford, Lander and Voyer, 2002; Aven and Ford, 2004; Taneja *et al.*, 2010; Asghari, Kashani and Hsu, 2021). ROA (Dixit and Pindyck, 1994) usually provides a boundary level for a stochastic phenomenon (e.g., state of the assets, fluctuations in costs) that triggers a specific type of action such as the implementation of a new project, starting maintenance and rehabilitation or even abandoning a project. The main drawback of ROA analysis lies within its burdensome dimensionality expansion. Even adding one more source of uncertainty to the conventional ROA makes

reaching the flexible plans and using them in practice arduous. A relatively recent line of studies has attempted to incorporate flexibility in IAM frameworks by formulating the asset management optimization problem as Markov Decision processes (MDPs) and adopting approaches to solve the mentioned MDP problem (Andriotis and Papakonstantinou, 2019, 2021; C. Wang *et al.*, 2020; Yao *et al.*, 2020; Yehia, 2020; Ou, Chang and Chakraborty, 2021). MDP formulation makes adding various sources of uncertainties to the problem possible.

MDP and achieving corresponding optimal solutions have long been a target of researchers in a variety of domains (Silver *et al.*, 2016; Khadilkar, 2019; Park *et al.*, 2019; Serrano, 2019; Renard, Corbett and Swei, 2021). Computer scientists and mathematicians have proposed a series of algorithms and methods for solving MDP and categorized them as a family of artificial intelligence (AI) called reinforcement learning (RL). RL methods aim to train AI agents for selecting optimal strategies through repetitive interaction with an MDP environment ("game" or "microworld") and observing the outcomes of the selected actions. RL methods have shown excellent potential for providing solutions for MDP problems with different inherent characteristics, outperforming conventional and classical methods, and surmounting human performance. AlphaGo is perhaps one of the most famous examples of such attempts (Silver *et al.*, 2016). In 2016, researchers and scientists at Google developed an intelligent AI with RL algorithms to play a classic yet complex board game, GO, without prior knowledge of the game. The trained agent could defeat a world champion GO player in several rounds (Silver *et al.*, 2016). Given their capabilities in dealing with complex problems with a large number of uncertainties, RL methods have also been used in automating industries, robotics, healthcare, marketing, and engineering fields for handling operational uncertainties and multi-objective procedures (Chembe *et al.*, 2019; Shi *et al.*, 2019; Valladares *et al.*, 2019; Ye, Li and Juang, 2019; Memarzadeh and Pozzi, 2019; Serrano,

2019; Brandi *et al.*, 2020; Zou, Yu and Ergan, 2020; Krishna Lakshmanan *et al.*, 2020; Liang, Kamat and Menassa, 2020; Si *et al.*, 2021; Xia *et al.*, 2021; Erharter *et al.*, 2021; Ghannad, Lee and Choi, 2021; Hodge, Hawkins and Alexander, 2021; Mushtaq *et al.*, 2021; Renard, Corbett and Swei, 2021). It is expected that RL-based methodologies, both in practice and academia, continue to grow in the coming decade (Nguyen and La, 2019; Shin *et al.*, 2019; Vázquez-Canteli and Nagy, 2019; Nian, Liu and Huang, 2020).

Project managers, decision-makers, and stakeholders in CEM are not unfamiliar with data-driven simulation and data analytics frameworks. Various AI methods, such as symbolic reasoning and statistical ML methods, have been supporting data-driven simulation and data analytics frameworks for supporting CEM decision-making. For nearly decades (Xu *et al.*, 2021), supervised ML algorithms (e.g., convolutional/deep neural networks (DNN), support vector machines (SVM), decision trees (DT), and random forests (RF)), and unsupervised ML methods (e.g., K-means and Density-based spatial clustering of applications with noise) have been boosting the study and analysis of complex phenomena and decision makings, such as image recognition, computer vision applications, cost estimation, cost forecasting, behavioral analysis, infrastructure systems fault detection, and safety science domains in CEM projects (Socher *et al.*, 2011; Tabandeh and Gardoni, 2015; Ashkan *et al.*, 2020; Chen *et al.*, 2020; Chen, Leng and Labi, 2020; Sai, Sanayei and Smith, 2021; Wang and Tang, 2021). Recently, RL methods, because of their scalability, generalizability, and rather simplicity, have attracted the attention of a growing body of CEM researchers in different domains for solving MDPs formulated problems such as bridge management (Papakonstantinou and Shinozuka, 2014; Andriotis and Papakonstantinou, 2019, 2021; Wei, Bao and Li, 2020; Khazaeli, Nguyen and Goulet, 2021), pavement management (Yao *et al.*, 2020; Yehia, 2020; Renard, Corbett and Swei, 2021), buildings (Chen *et al.*, 2018; Jung *et al.*, 2021), and infrastructure resilience (Memarzadeh and Pozzi, 2019; Nozhati, Ellingwood

and Chong, 2020; Ghannad, Lee and Choi, 2021; Lu, Xuan and Sunil, 2021). A more detailed review of such attempts in the CEM domain can be found in the literature review chapter of this dissertation.

Application of RL in the IAM problem, similar to other domains, can provide the potential for 1) adding flexibility to, 2) considering several sources of uncertainties during the training and analysis of, 3) reaching potentially more beneficial solutions for the IAM decision-making process. The RL-based IAM studies have taken important steps and paved the way toward introducing the notion of RL to the IAM domain and community. Acknowledging the efforts made by these studies on developing the theoretical and practical foundations of RL application in IAM, literature analysis of the RL-based IAM studies shows that some limitations can still be found in all or some of them. First and foremost, hazards, assets' response to them, and corresponding loss costs seem not to have gained proper attention in these RL-based studies. Second, future uncertain changes in costs, which play crucial roles in the decision-making process, have not been considered in these studies. Third, despite the sound theoretical foundations shaped by these studies, long-established components of IAM frameworks, such as user costs, standard condition rating (CR) of elements proposed by departments of transportation, and utility theory, seem not to have received appropriate considerations. All in all, IAM literature needs a holistic asset management framework consisting of the long-established and conventional components of IAM as well as consideration of unprecedented hazards, volatile market conditions, and network-level analysis.

Network-level infrastructure asset management (NL-IAM) methodologies aim at optimally selecting a subset of viable actions from a portfolio of intervention actions for assets in a network (Thompson *et al.*, 1998; Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011). NL-IAM has been formulated as an optimization problem in a variety of ways such as

the multichoice multidimensional knapsack problem (Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011). The obtained solutions determine exactly one optimal action (including do nothing) for all (elements of) assets in a network. This optimization problem is fairly complex. The complexity stems from the curse of dimensionality and multiple financial and resource constraints (Andriotis and Papakonstantinou, 2021). Due to these complexities, long-established methods and relatively new solutions can only be achieved by simplifying various aspects of this optimization problem (Patidar, Labi, Morin, Paul D Thompson, *et al.*, 2011; Alireza, Luis and Fuzhan, 2020; Hou and Ai, 2020; Jiang, Wu and Wu, 2021; M., Oscar and Jeffrey, 2021). However, the simplification process imposes several limitations (e.g., inflexibility of the decision-making process and excessively simplified decisions in face of complex unprecedented uncertainties) to both the practice and research in this domain.

So far, however, little attention has been paid to extending RL application into NL-IAM. In one study, deep Q-networks (DQN) were adopted for long-term pavement maintenance planning of pavement sections (Yao *et al.*, 2020). It focused on different sections of two expressways and showed the applicability of DQN in long-term decision-making. This study focused on section-level and proposed ranking and selecting the actions for different sections based on their Q-value within the budget constraints. Acknowledging the upsides of this approach, it is associated with two main drawbacks. First, DQN and deep Q-learning are heavily impacted by the deadly triads (Sutton and Barto, 2018) and catastrophic forgetting (Kirkpatrick *et al.*, 2017) which make the learning unstable and consequently the results unreliable. Another, yet important, drawback of this approach lies in the negligence of possible optimal combinations of concurrent actions that could not be derived by selecting them separately. Another recent study put forward a deep decentralized multi-agent actor-critic model for long-term inspection and maintenance planning of a bridge with 10 elements

in its life cycle. This study meticulously addressed the curse of dimensionality by function parametrization, curse of history, and state uncertainty issue by formulating the IAM problem as a partially observable MDPs, and different types of constraints by constraint control and Lagrangian relaxation. Acknowledging the solid mathematical justification and credibility of such solutions, we claim that these solutions might not be equally effective when a large number of states and actions (i.e., assets/elements and their corresponding actions) exist in the problem. In other words, their implementation would lead to infeasible computational costs for a small network with as few as 50 assets and 150 elements with 600 different actions.

1-1 Objectives

Enhancing and improving the current practice of asset management systems from availability, computationally, and flexibility are the major objectives of this dissertation. First, an open-source and extensible asset management software is developed and introduced. Then, ML-based methodologies for reducing the computational time of both analysis loops of the commonly used MCS-HOA framework are put forward. Intending to add flexibility to the decision-making process both at the PL-IAM and NL-IAM, the rest of this study focused on providing a framework, guidelines, and solutions for RL-based PL-IAM and NL-IAM. Overall, the general objectives of this dissertation can be summarized as follows:

1. Developing a general, extensible, and open-source infrastructure asset management software
2. A methodology for learning the behavior of Monte Carlo simulation by machine learning models for estimating the results of life cycle costs analysis under uncertainties

3. A methodology for learning the behavior of optimization algorithms by machine learning models and predicting the optimal maintenance planning of assets under uncertainties
4. Developing a holistic framework for adopting reinforcement learning methodologies and utilizing the long-established aspects of infrastructure asset management under uncertainty in reinforcement learning-based studies
5. Network-level infrastructure asset management under complex uncertainty using reinforcement learning methods and developing recommended solutions for the curse of dimensionality and reward engineering

1-2 Overview of the dissertation's methodology and case studies

This dissertation is summarized in Figure 1.1 and Figure 1.2 and in the following paragraphs. Chapter 2 briefly summarizes the background of this research including topics such as open collaboration, previous investigations, and efforts for reducing the computational time of the commonly used IAM systems, the current state of RL methods in CEM, and corresponding discussions.

Following a review of previously proposed studies on various IAM systems, open-source ethos in CEM and civil engineering, and bridge management systems, Chapter 3 introduces most modules and models, that a general IAM framework might contain as a general architecture for IAMs, as well as their implementations in the developed software GIAMS. These modules on a high level include the asset module, the network module, the life cycle analyzer module, optimizers, and report generators. Then, A BMS is developed based on the Indiana network of bridges is formed for further analysis in the whole dissertation. All the models and data, such as the condition rating definition (FHWA (Federal Highway Administration), 2012), the condition and characteristics of the bridges in the Indiana

network (National bridge inventory, US), deterioration models, MRR models, agency costs (Sinha *et al.*, 2009), hazard models (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003), hazard data (USGS, 2020), user costs models (TexasDOT, 2020), and utility models (Li and Sinha, 2004) were adopted from previous studies. Apart from these models, a set of other models were adopted from some other models such as effectiveness models and recovery models were inspired by previous studies (Hawk and Small, 1998). All chapters and methodologies were showcased in this example. Then, a project-level IAM and a network-level IAM are provided to show the applicability of the proposed platform and the developed examples. Further details of the models are provided in Chapter 3. In addition, this chapter introduces the techniques and some examples for open collaboration and extension of the developed software.

Chapter 4 focuses on developing a methodology for estimating the LCCA results using DNN. This methodology comprises a certain set of steps including formulating the LCCA, parametrizing the LCCA, synthesizing datasets for training models, training ML models, and validating the results. The case study in this chapter is based on formulating and parametrizing the LCCA module provided in the developed and designed example in Chapter 3. Then, the LCCA procedure was formulated and parametrized followed by synthesizing millions of samples containing synthesized assets, random MRR models, and simulation parameters. Then, the datasets were undergone standard data preprocessing processes such as feature encoding, feature selection, and feature scaling. Next, DNN models were trained and validated on the synthesized data.

Chapter 5 takes advantage of the trained DNN models in the previous chapter and puts forward an ML-based methodology to directly predict the outcomes of the MCS-HOA framework without doing the simulation or optimization. This methodology in nature is

similar to the methodology provided in Chapter 4. In other words, first, the PL-IAM was formulated and parametrized before finding optimized MRR plans of synthesized assets with random simulation parameters. After similar preprocessing processes such as feature selection, feature encoding, and feature scaling, an ensemble of random forests models were trained and validated on the data for learning the behavior of GA and predicting the optimal MRR plans.

Chapter 6 looks at the IAM problem from another perspective and intends to develop a holistic guideline and framework for adding flexibility to the IAM problem. This chapter starts by introducing the MDPs problems and continues to formulate IAM as MDPs. Then, a general approach to converting IAM systems to microworlds is provided as well as the microworld based on the provided example in the dissertation. Then, DQN and A2C multi-agent RL models were designed and developed based on the elements of bridges in the main example of the study. Next, the RL models that were designed based on the bridge in the PL-IAM example in Chapter 3 were trained and the final results were validated against the optimal MRR plan and by visual investigations.

Chapter 7 presents one of the first NL-IAM analyses and solutions for tackling the curse of dimensionality and reward engineering in the adopted RL method. Technically speaking, multi-agent A2C models were designed, developed, and trained on a network of bridges for flexible and optimal decision making. An innovative and novel network-level IAM system using the developed DNN models for estimating LCCA results and RF models for estimating the optimal MRR plans based on incremental utility costs ratio algorithm were used for verifying the results of the RL agents.

This dissertation is ended with a summary of the contributions and findings of this dissertation as well as opportunities for future research.

	Literature review	Methodology keypoints	Case studies	
Chapter 3	❑ Various IAM systems ❑ Open source ethos in CEM ❑ Previously developed BMSs	The GIAMS' architecture ❑ Design ❑ Develop • Assets • Network • Life cycle analyzer • Optimizers • Report generators	❑ A BMS based on the Indiana network [The base example for next sections] • Condition rating: (FHWA, 2012) and NBI • Deterioration models: (Sinha et al., 2009) • MRR Models: (Sinha et al., 2009) • Agency costs: (Sinha et al., 2009) • Hazard models: USGS and (FEMA-NIBS, 2003) • User costs models: (TexasDOT, 2020) • Utilities models: (Li and Sinha, 2004)	❑ Optimize the life cycle MRR planning of a bridge ❑ Optimize the MRR plans for a network based on a pre-defined portfolio of actions
Chapter 4 & 5	❑ Importance of computation time ❑ Previous studies for accerlating the IAM systems	❑ Formulate the LCCA ❑ Parameterize ❑ Synthezie datasets ❑ Train ML models ❑ Validate	❑ Formulate and parametrize the LCCA in the case study ❑ Synthezie datasets for LCCA by conducting millions of MCS analysis ❑ Pre-processing: Feature selection, encoding and scaling ❑ Train DNN models to learn MCS behavious and predict MCS results ❑ Validate the predictions	
		❑ Formulate the PL-IAM ❑ Parameterize ❑ Synthezie datasets ❑ Train ML models ❑ Validate	❑ Formulate and parametrize the PL-IAM in the case study ❑ Synthezie datasets by conducting numerous PL-IAM using GA and trained DNN models ❑ Pre-processing: Feature selection, encoding and scaling ❑ Train ensemble of RL models to learn GA behavior and predict PL-IAM results ❑ Validate the predictions	
Chapter 6 & 7	❑ RL in CEM ❑ RL in IAM	❑ Develop the microworld ❑ Setup the RL models ❑ Trail RL agents ❑ Find a baseline ❑ Validate results	❑ Develop the microwrold based on the case studies in previous steps ❑ Setup DQN and A2C RL agents for different elements of the bridge example in chapter 3 ❑ Train RL agents to find optimal flexible strategies ❑ Use the optimized MRR plans derived by GA as the baseline ❑ Verify and analyze results	
		❑ Setup the RL models for network level analysis ❑ Trail RL agents ❑ Find a baseline ❑ Validate results	❑ Setup multi-agent A2C models for network level analysis ❑ Developing solutions for curse of dimensionality ❑ Train RL agents to find optimal flexible strategies for a network ❑ Use the trained DNN and RF models with IUC algorithms to find a baseline ❑ Verify and analyze results	

Figure 1.1. An overview of the methodology and the whole dissertation sections

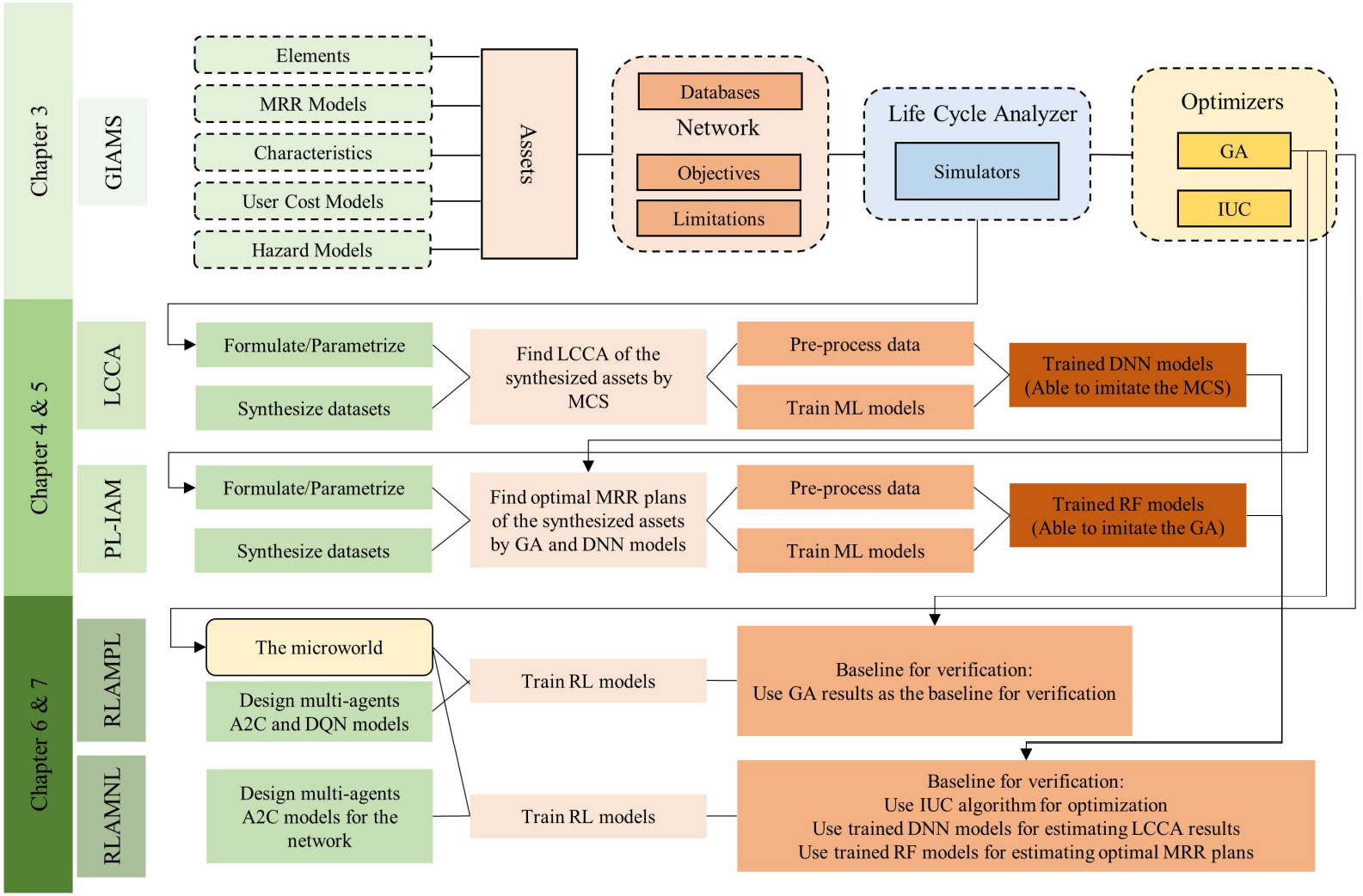


Figure 1.2. Summary of the models used in the dissertation

Chapter 2 Literature review

This chapter focuses on providing: 1) an overview of previous studies on open-source computer programs in civil engineering and open collaboration ethos are provided, 2) an overview of the importance of computation time in IAM and previous efforts in the literature for enhancing IAM systems in terms of computational time, 3) and overview of and discussion on RL application in CEM domains and IAM.

2-1 Open collaboration and open-source software in CEM

2-1-1 Open-source civil engineering computer programs

Several computational programs for civil engineering problems have been developed in the past decade. For example, Mahsuli and Haukaas (2013) developed *Rt*, a freely available computer program, for reliability analysis and probabilistic modeling with a focus on the seismic analysis of different structures. This computer program has been under development and upgrading since its first version. Pagani et al. (2014) developed an open-source platform called *OpenQuake* engine for risk calculation and hazard analysis of earthquakes globally. OpenQuake is also currently under development and extension with open collaboration on GitHub. *Optimist* is another open-source Python library that was developed by Raso et al. (2016) to ease stochastic dual dynamic programming in water systems operation and analysis. Warren et al. (2016) developed another open-source software for simulating the electromagnetic wave propagation in soil by numerical models. Gadi et al. (2020) proposed a Python program for image processing to evaluate soil moisture content. *OpenSeesPy* was developed by Zhu et al. (2018) for finite element analysis of structural and geotechnical systems and their response to seismic hazards in Python. Guan et al. (2020) developed another computational Python-based platform called *AutoSDA* to facilitate the seismic design,

modeling, and analysis of steel moment frames. Similarly, attempts have been made and other Python-based platforms have been developed in the structural engineering area (e.g., structural health monitoring (Jagadale *et al.*, 2020), materials resistance analysis (Rivera S and Estupiñán L, 2020)). *PySWMM* was developed by McDonnel et al. (2020) to model and analyze stormwater and related phenomena in Python. Zeraoui et al. (2020) developed another program for the optimization of various processes in managing dredged sediments. Developing computer programs, and more specifically open-source and Python-based platforms have gained more attention in the past few years. This philosophy, however, is yet to become popular and widely used in civil engineering, and more specifically in the CEM field.

2-1-2 Open collaboration

The open collaboration approach to software engineering has been in practice for years. Perhaps one of the first and greatest examples of open-source projects is the Linux kernel (Torvalds, 2020), which is used in billions of devices around the world. More than ten thousand collaborators, most of whom have no direct acquaintance with each other, have contributed to this project since the early 90s. In open collaboration, contributors develop valuable features, reuse, and share each other's output, work purposefully toward a common goal, and permit anyone to contribute and consume (Levine and Prietula, 2014). Such an ethos enables collaborators around the world to directly build upon each other's work, interact with each other, and share their knowledge and resources. Consequently, it accelerates the speed of developing any item, lowers costs, and opens the gate for new advancements. A similar approach has already been adopted in most Python libraries and packages for scientific and industrial purposes (Pedregosa *et al.*, 2011; Chollet, 2015). So far, however, there has been little effort devoted to providing an open-source asset management

system to be employed both in research and practice. The developed framework in this dissertation intends to fill this gap.

2-2 Importance of computation time and effort in reducing computation time of IAM systems

2-2-1 The importance of computation time and the use of simplified models

Both PL-IAM and NL-IAM must be conducted within a workable computation time so that the relevant agencies and decision-makers can analyze and evaluate a multitude of MRR alternatives (Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011). The importance of low computation time has been raised in several studies (Kandil, El-Rayes and El-Anwar, 2010; Frangopol, 2011; Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011; Yang, Hsieh and Kung, 2012; David Y. Yang and Frangopol, 2020a). To bypass this obstacle, NL-IAM systems that are conducted as part of real-life decision-making tools have typically used simplified and deterministic models (e.g., IBMS (Sinha *et al.*, 2009)). These models have been used to analyze complex phenomena governing the asset (e.g., non-probabilistic deterioration models (David Y. Yang and Frangopol, 2020a)), provide pre-defined MRR projects (e.g., DTREE in the Indiana bridge management system (IBMS) (Sinha *et al.*, 2009)), and select the best subset of projects by heuristic project selection methods (e.g., incremental utility costs (IUC) ratio heuristics (Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011)). While simplified models developed in previous studies have played a vital role in practical asset management to date, they are by design not able to fully capture complex natural and environmental phenomena (Sánchez-Silva *et al.*, 2016).

2-2-2 Computational time reduction in IAM systems

Given the importance of computation time in asset management, past studies have focused on upscaling advanced complex asset management frameworks to enhance their decision-

making process. Yang, *et al.* (2012) proposed parallel computing for improving the computation time of multi-objective optimization (MOO) using particle swarm optimization (PSO) and MCS for a bridge management system (BMS). Their proposed methodology could yield the analyses' results up to 60 times faster. Multi-attribute utility functions have also been incorporated in MOO problems to reduce dimensions, and consequently computation time, of asset management problems (Bai *et al.*, 2013; Kim and Frangopol, 2018). Despite the improvements made by these studies, the development of an NL-IAM system based on an LCCA and optimization of assets without compromising complex models has been a challenge (Frangopol, 2011), and has remained so heretofore.

This means the majority of conducted research in PL-IAM taking into account complex models (e.g., probabilistic deterioration models, uncertain hazard and response models, and costs models with volatilities) are still far from applicability to all assets in a network. Therefore, there is a need for a methodology that can reduce the LCCA computation time (the first computational loop of the MCS-HOA framework) and the HOA optimization times (the second computational loop of the MCS-HOA framework) to make much-needed headway in NL-IAM without compromising complex models (i.e., non-linear or probabilistic models). ML models, such as DNN, are characterized by their capabilities in learning complex correlations in a variety of systems. They have been widely used in different domains of knowledge for developing managerial and decision support frameworks, such as disaster assessment in transportation engineering (Yudi, Qi and Wenying, 2020), hazard assessment of dam infrastructure (Rayan and H., 2020), construction costs of infrastructure assets (Ilker, D. and David, 2020), project management and planning (Mohamad, Jordan and M., 2021), and implementing renewable energies in communities (Jackson *et al.*, 2020), in the past few years. This dissertation utilizes ML models to train models capable of estimating LCCA and optimization results

2-3 Reinforcement learning in CEM and IAM

Given its similarity to human learning, ability to solve problems with a multitude of different uncertain states, learning capability by interacting with an environment without a need for large datasets, RL has long been adopted in various domains for handling interactions between decision agents that are performing parallel or interwoven workflows with various uncertainties, either on the agent behaviors or on the changed environmental conditions. In fact, some studies found that RL can outperform traditional optimization, decision making, and search methods defeating human-level intelligence and experience.

Compared with these existing RL applications, CEM applications involve highly uncertain human behaviors embedded within decision processes and field workflows. While un/supervised algorithms are capable of learning and conducting single-stage decision-making problems, RL has shown the potential for handling dynamic and uncertain sequential decision-making scenarios (known as MDPs) in various domains of science. The full potentials and extent of applications of RL in both CEM practice and research are yet to be fully exploited. RL methods can unleash new opportunities and improve managerial decision-making frameworks in the CEM domain where decision contexts have various uncertainties associated with parallel workflows and interacting behaviors between human, machine, and information systems.

2-3-1 Classification of applications domain

RL algorithms intend to maximize a pre-defined reward by making optimal decisions under a variety of circumstances in a limited or unlimited time span. This definition makes the RL algorithm potentially applicable to a variety of decision-making processes in uncertain environments involving multiple interactive decision-making agents and environmental

processes. Based on the reviewed papers, the following passages provide brief introductory discussions about the current state, strengths, and limitations of RL-based studies that are directly or indirectly related to CEM projects. Examples of such problems include project time and costs coordination under uncertainties, balancing throughput of an airport while minimizing operational risks, and machinery operation on construction sites. (Sun *et al.*, 2019, 2020). Although these categories differ in their application domains, they share a similar set of characteristics: they all can be formulated as MDP problems. To illustrate, in all these CEM domains, a sequence of actions need to be taken given the state of the problem under various current and future environment uncertainties to maximize/minimize long-term non-/deterministic rewards.

The application of RL algorithms in CEM can be classified into the following domains: 1) building energy and management, 2) building heat, ventilation, and air conditioning (HVAC), 3) infrastructure management, sustainability, and resilience, 4) construction machinery, robots, and tools, 5) design, 6) manufacturing and construction processes, 7) project scheduling and resource allocations, 8) others. The number of articles related to each category with their references is provided in Table 2.1. The next sections of this chapter provide more information about the details of past studies focusing on IAM.

Table 2.1. Categories of collected RL-based papers given their application domain

Domain	Abbr.	Num.
Building Energy Management and HVAC control	Ener.	51
Infrastructure Management, Sustainability, and Resiliency	Infra.	14
Construction Machinery, Robots, and Tools	Mach.	9
Design	Design	6
Others	Others	5

2-3-2 Classification of methods

The characteristics of state and action space heavily affect the designs of different elements of an RL approach such as the value approximation function, optimization methods, and mapping states to actions. For example, the Q-learning algorithm is designed for small problems with a limited number of states and actions and is not suitable for problems with continuous action space. Some of the well-known methods with their characteristics as well as their corresponding example in CEM are summarized in Table 2.2.

To gain further knowledge on the distribution of the RL methods among the reviewed papers, Figure 2.1 depicts the pie chart of all employed RL approaches. It should be noted that some studies used a trivial variation of the common groups of methods depicted in Figure 2.1. They are merged into their bigger families. For example, Memarzadeh and Pozzi (2019) used safe Q-learning in their study but we counted it as the DQN method. This was mainly done to provide a more informative figure rather than a pie chart with multiple slices corresponding to methodologies with one occurrence among the collected papers. Based on this figure, Q-learning (either Tabular or with linear approximation) was the most used methodology followed by DQN and actor-critic methods. These algorithms are associated with different strengths and limitations. A rather thorough discussion on this matter is provided in the following.

Table 2.2. A summary of the characteristics of RL approaches with examples

RL approach	On/Off policy	State space	Action space	Value function estimation method	Example
Monte Carlo (MCTS)	Both	Discrete	Discrete	Sample means	Maintenance planning of assets under different uncertainties
TD-lambda (TD)	Both	Discrete	Discrete	Sample updates with dynamic programming	
Tabular Q-learning (Q-learning)	Off- policy	Discrete	Discrete	Q-value	
SARSA	On- policy	Discrete	Discrete	Q-value	
(Double) DQN	Off- policy	Continuous	Discrete	Q-value	
Policy gradient (PG)	Off- policy	Continuous	Continuous	Q-value	
Deep deterministic policy gradient (DDPG)	Off- policy	Continuous	Continuous	Q-Value	Building HVAC control given various states
Actor critic (A2C, A3C)	On- policy	Continuous	Continuous	Advantage	
Proximal policy optimization (PPO)	On- policy	Continuous	Continuous	Advantage	

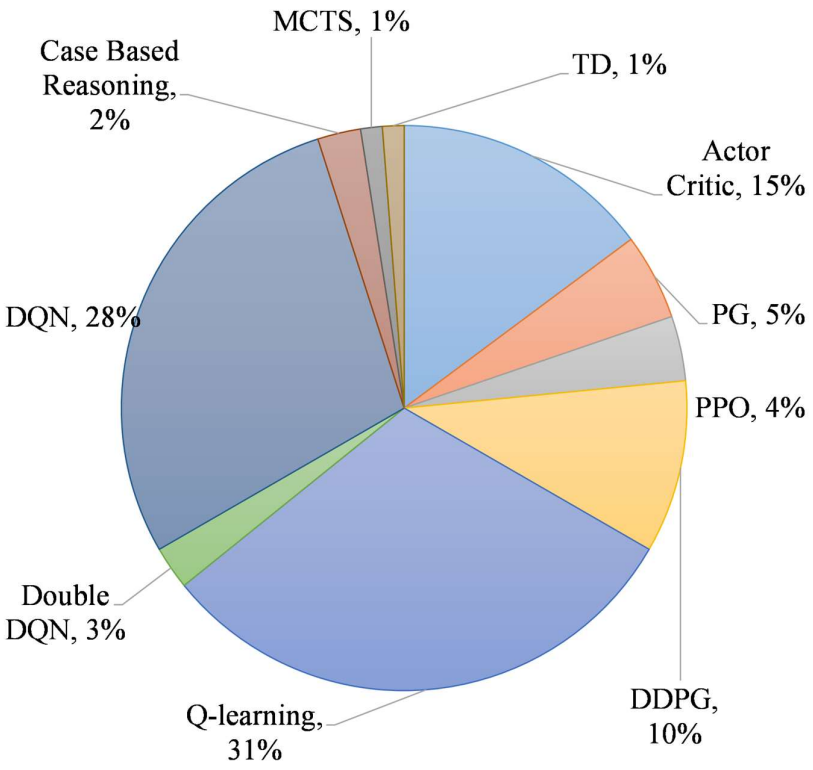


Figure 2.1. Common RL methods used in the reviewed studies (n=85)

2-3-3 RL in IAM, sustainability, and resiliency

With fourteen articles among the collected papers, the infrastructure management, sustainability, and resiliency categories rank third among all categories. Be it post-hazard community resiliency, structural health monitoring, or long-term life cycle optimization of assets, a sequence of actions needs to be taken in the management horizon of asset(s) or a community to minimize factors such as costs, global warming, risks while maximizing other factors such as resiliency and performance. The extent of these actions needs to be properly adjusted given budgetary limitations and uncertain environmental factors, such as the changing condition of elements and assets and varying market conditions. This problem has been usually formulated as a Markov Decision Processes (MDP) (Papakonstantinou and

Shinozuka, 2014; Andriotis and Papakonstantinou, 2019, 2021). RL has recently gained popularity to solve this challenging decision-making problem.

In one of the earliest attempts toward utilizing RL in asset management, Durango-Cohen (2004) proposed a temporal difference approach for maintenance planning of infrastructure facilities. The purpose of this study was to eliminate the Markovian deterioration models, the accuracy of which has always been doubted, from asset management studies. The results of the proposed temporal-difference learning model were promising in comparison to the maintenance planning as a result of the optimal policy, SARSA, and Q-learning. Almost ten years later, researchers started a series of studies focusing on the partial observability of future uncertain events. In one of the first studies in this area (Papakonstantinou and Shinozuka, 2014), the authors tried to solve the MDP problem (lifecycle optimization of assets) with approximate solvers such as Perseus. Then, Andriotis and Papakonstantinou (2019) proposed a deep centralized multi-agent actor-critic (DCMAC) for a large state-action space of a system with multiple components. This work was the first study that used the notion of RL for the maintenance planning of a bridge. Later, Andriotis and Papakonstantinou (2021) designed a multi-agent actor-critic model for obtaining optimal inspection planning with incomplete information. Taking several types of constraints such as risk and budget constraints, they tried to gain a deeper understanding of optimal inspection policies for a 10-component deteriorating system (bridge) in a 50-years management horizon. Bridge maintenance and health monitoring were the targets of other studies related to infrastructure and asset management. (Wei, Bao and Li, 2020) have proposed a DRL-based methodology for component-level maintenance of one relatively simple and one complex bridge. Although their asset management framework on high levels is similar to the practice of bridge management, they came up with a new idea about encoding components' states. To illustrate, the conditions of elements were encoded and converted to 2D figures in their study.

Then, the 2D figures played the role of states within the RL framework. Next, a convolutional neural network model uses 2D-based state representations to approximate Q-values. In another study, Khazaeli et al. (2021) focused on structural health monitoring of bridges. They developed a framework consisting of a Bayesian Dynamic Linear Model (BDLM) and a Q-learning model to detect anomalies in a bridge. Using the collected data from the elongation and temperature of a bridge in Canada within a period of 6 years, they showed that their trained model can accurately detect anomalies and neglect false alarms.

RL in infrastructure management has not been limited to bridges. In a detailed-oriented study, Yao et al. (2020) developed a DQN framework for long-term maintenance planning of pavements. The authors used the data of two expressways in China to develop the states with 42 features (e.g., different types of pavement condition indicators) with 38 different actions. This study was different from other asset management studies from one point of view. Instead of using Markovian transition matrices for representing pavement conditions deteriorations, this study used trained neural networks models for performance prediction of pavement sections. They also offered a heuristic solution for extending this framework to the NL-IAM studies and practices. In short, the actions with higher Q-values within the budget limitations should be chosen. Recently, researchers from the University of British Columbia shifted the focus toward global warming. Motivated by the need to defeat the curse of dimensionality in optimization problems and to simultaneously consider several uncertain factors in one life cycle assessment (LCA) framework, Renard et al. (2021) developed a Q-learning framework for minimizing the global warming impacts of pavement maintenance actions in a 50-year horizon. They showed that the trained agent could outperform the classic methods by 18% to reduce emissions. Relaxing the Markovian property assumption of pavements' components was another highlight of this study.

Apart from the vast realm of asset management, community resilience has also been the focus of some RL-based studies. Nozhati et al. (2020) published a study about the potential of applying RL and approximate dynamic programming (ADP) to community resilience planning problems. In this study, the authors covered common practices in both RL and ADM and evaluated their usefulness for the aforementioned purposes. They suggested that Q-learning methods could lead to optimal values albeit with high computational costs. Instead, they highlighted the usefulness of the rollout algorithm for community resilience planning with large state-action spaces. A recent study (Ghannad, Lee and Choi, 2021) investigated the effectiveness of multi-agent RL (MARL) in post-disaster recovery actions of a city under two hurricane scenarios. They carefully modeled all details of hazards, recovery actions, and financial and socio-economic impacts of all parts of this microworld. The results of this study show that multi-agent reinforcement learning (MARL) applications could be extended to the resiliency of communities as well. Motivated by the need for a unified system to aggregate stakeholders' values after a disaster, Lu et al. (2021) proposed a q-learning approach toward ranking different strategies given to any state to maximize stakeholders' values for disaster resilience. The focal point of this study was proposing a new stakeholders' values aggregating model, using a hypothetical case study considering building structural robustness, community responses (e.g., evacuation efficiency, safe and healthy nationhood, and natural ecosystem protection), and market condition (i.e., high return on investment) for taking the optimal actions. These studies have taken steps toward the application of RL in community resilience, though further studies are required before their practical applications.

Apart from these studies, a few other studies were relevant to the infrastructure management, sustainability, and resilience category but could not be allocated to the previous groups of studies. These studies tried to provide solutions for domain adaptation of Q-learning (i.e., safe Q-learning for costly infrastructure management systems decisions

(Memarzadeh and Pozzi, 2019)), develop a self-monitoring and repair system in buildings (Serrano, 2019), and propose a new sampling technique in reliability analysis by RL techniques (Xiang *et al.*, 2020).

2-3-4 Discussion on the application of RL in CEM and IAM

This section discusses the limitations of previous RL-based studies. Some of the limitations of the reviewed papers are already addressed by their authors, yet some of them seem to need further studies. It should be noted that although some of the following discussions might merely address one category of CEM applications, they usually hold for other CEM applications such as IAM as well.

2-3-4-1 On the justification for using RL

Development, design, and formulation of a microworld as well as hyperparameter tuning and training of RL agents take a great deal of effort and time. As a result, RL-based approaches are only desirable if: 1) the problem is overly complex and optimized solutions cannot be reached 2) the solutions would be achieved in an unfeasible amount of computation time. Feasible computation time highly depends on the type of problems and applications. Such time could range from milliseconds for real-time decision makings in automated vehicles to days and weeks for training intelligent infrastructure management systems. Since trained RL agents usually make decisions via neural networks, the computation time of decision-making is usually milliseconds. This makes trained RL agents more favorable than other optimization algorithms with more significant computation time. However, the development, hyperparameter tuning, and training time of RL agents should also be counted in these contexts. In other words, if perfect or near-perfect solutions could be achieved by simpler methodologies such as brute force algorithm, RL agents trained for one single purpose/problem are less useful.

2-3-4-2 On generalizability of the examples

RL-based methodologies should be applied for deriving optimal or near-optimal strategies in problems where similar solutions cannot be achieved faster. Take the work of Sahachaisaree et al. (2020) as an example. Despite the fact that their results showed significant improvement in terms of computation time compared with the classic optimization problems, the scalability of such an approach to real-life problems with thousands and even millions of states and options is problematic. These studies (e.g., (Ruiz-Montiel *et al.*, 2013; Krishna Lakshmanan *et al.*, 2020; Liu *et al.*, 2020; Mandow *et al.*, 2020; Shi *et al.*, 2020)) are all in their infancy stages; however, the scalability and generalizability of RL methodologies would need extensive discussions in all RL-based studies. Spending hours and days on the development and training of a single-purpose AI agent is practical if and only if the long-term benefits of finding such optimal solutions are bigger than the overall costs of training the agent.

2-3-4-3 On the network-level analysis

The scalability and generalizability of a trained agent take the form of network-level analysis in the context of infrastructure management and maintenance planning. Although multiple studies with different methodologies focused on the project-level analysis of infrastructure assets (i.e., maintenance planning, structural health monitoring optimization, and seismic upgrade of structures), stakeholders are usually responsible for a network of assets. In network-level analysis, stakeholders and decision-makers have to select a subset of possible projects to maximize pre-defined utilities. All studies in the infrastructure, sustainability, and resilience category, except the work of Yao et al. (2020), did not mention nor provide recommendations for network-level analysis. Even the solution proposed by Yao et al. (2020), which is selecting projects with the highest Q-values within budget limitations, needs further investigation as this suggestion is based on the validity of non-linear reward functions'

additivity property. Not to mention, this concept has already been investigated thoroughly (Bai *et al.*, 2013).

2-3-4-4 On validation of results

One key step in conducting academic research is the validation of results. The history of academic research contains some non-validated research results leading to catastrophic consequences (e.g., (Eggertson, 2010)). CEM-related RL-based studies, such as IAM problems, are no exception. Nonetheless, several studies in the survey focused on pushing the boundaries of their research fields and contained relatively fewer results on thoroughly validating the RL solutions(e.g., (Adam and Smith, 2008; Papakonstantinou and Shinozuka, 2014; Serrano, 2019; Valladares *et al.*, 2019; Andriotis and Papakonstantinou, 2019; Liu, Cao and Lei, 2019; Akanmu *et al.*, 2020; Wei *et al.*, 2020; Mullapudi *et al.*, 2020; Nozhati, Ellingwood and Chong, 2020; Erharter *et al.*, 2021; Jung *et al.*, 2021; Lee and Kim, 2021)). Like some other control methods and optimization algorithms, RL methods can converge to any point or strategy. Even if the simulation engine is problematic or the training code is faulty, RL agents could still converge to some points and strategies. Therefore, it is pivotal to validate the results in a professional setting. HOA(e.g., in (Krishna Lakshmanan *et al.*, 2020)), rule-based strategies (e.g., in (Chemingui, Gastli and Ellabban, 2020)), human performance (e.g., in (Lee and Kim, 2021)) are some of the baselines with which the trained agents could be, and perhaps in the future research should be, validated in a more comprehensive and systematic manner.

2-3-4-5 On downsides of Q-learning and DQN

Despite the usefulness of deep Q-learning (or DQN) algorithms, they are associated with multiple limitations. In their renowned book, (Sutton and Barto, 2018) introduced the deadly triad of RL algorithms, highlighting 1) the instability of RL methods when 2) function

approximators (such as neural networks) are used in 3) an off-policy setting with bootstrapping techniques. Simply put, Q values in DQN could easily diverge in even small problems with as few as five states. Studies are being undertaken to tackle this limitation (e.g., in (Zhang, Yao and Whiteson, 2021)); however, these methods are yet to become theoretically well-founded converging to optimal points. The downsides of DQN methods are not limited to the instability caused by the deadly triads. Q-learning and consequently DQN methods are also well-known and well-documented for overestimating the Q-values. Double Q-learning (Hasselt, 2010; Van Hasselt, Guez and Silver, 2015) and dueling DQN (Wang, Freitas and Lanctot, 2015) are some of the techniques that have been suggested to tackle the overestimation issue.

Another crucial drawback of DQN methods stems from the inherent nature of neural networks. In spite of the human brain, neural networks tend to forget previously learned and older tasks in time (i.e., after several new observations). This phenomenon has been referred to as catastrophic forgetting and previous studies. Kirkpatrick et al. (2017) have tried to overcome this problem of "catastrophic forgetting" by proposing weights importance to older tasks. Last, but not least, Q-learning and DQN methods usually converge slower than other methods (e.g., in (Liu and Henze, 2006; An *et al.*, 2021; Lissa *et al.*, 2021)) leading to infeasible training time in highly complex problems.

As shown in Figure 2.1, deep Q learning has been the most popular algorithm among CEM researchers so far. Unfortunately, most CEM-related studies have chosen to briefly mention the shortcomings of DQN and have not yet systematically studied how to address these shortcomings. Faulty or sub-optimal decision makings in the CEM problem could potentially lead to the loss of human lives and billions of dollars. Therefore, it is extremely

important that future CEM-related RL-based studies emphasize the stability and convergence of their selected RL-based methods.

2-3-4-6 On online availability of codes

The computer science and engineering field have long benefited from the notion of open collaboration (Levine and Prietula, 2014). Open collaboration makes the extension of previous works easier reducing massive amounts of energy from programmers and developers. This notion is relatively new in CEM (e.g., in (Asghari and Hsu, 2021)). A limited number of reviewed studies in this survey provided their codes and microworlds available online (e.g., (Liu, Cao and Lei, 2019; Mullapudi *et al.*, 2020; Apolinarska *et al.*, 2021; Bowes *et al.*, 2021; Erharter *et al.*, 2021). This would lead to hours of repetitive work and a waste of energy from researchers and practitioners. From our viewpoint, providing the source code of the simulation engines and RL codes in future studies could greatly benefit both the practitioners and researchers in CEM.

2-4 Bridge management

2-4-1 Importance of bridge management

Bridges, as one of the most important components of transportation systems, have gained particular attention in the asset management literature. That is mainly due to the fact that the economy, safety, and almost every aspect of life in modern communities heavily rely on transportation networks. In addition, bridges' failure can potentially lead to dire consequences. Transportation networks' components such as bridges are in critical condition (American Society of Civil Engineers, 2021). As of early 2021, for example, US' transportation network has approximately 260,000 bridges with more than half a century of age and 46,000 bridges in poor condition requiring immediate attention (American Society of Civil Engineers, 2021). In the US, for example, this concern has led to the passing of a

tremendously comprehensive bill for upgrading and maintenance of the transportation network (Cochrane, 2021). Due to their indisputably important role and their degraded states around the world (e.g., in (American Society of Civil Engineers, 2021)), BMSs have been the focus of hundreds of studies so far (Alysson, M. and Liang, 2018). Given the newly enacted funding and the current status of the transportation systems, it is expected that this line of research continues to grow and enlarge in the coming decade.

2-4-2 Famous bridge management systems

A famous bridge management system, PONTIS (Thompson *et al.*, 1998), was developed in the 1990s under the sponsorship of the Federal Highway Administration and has been widely used in the US. It provides a network-level bridge management strategy and optimizes a multitude of maintenance, rehabilitation, and reconstruction (MRR) projects. During the same period, and for the same type of asset, BRIDGIT (Hawk and Small, 1998) was developed for smaller networks. It could provide the decision-makers with both network-level and project-level analyses of the assets. PONTIS and BRIDGIT have long been used in several departments of transportation in the US and have been very practical. In the absence of highly sophisticated computers, these asset management systems could provide rational strategies based on theoretically-well founded methodologies. However, BMSs have been advancing both in practice and in the literature. Almost ten years later, for example, the IBMS received a major update (Sinha *et al.*, 2009). In this BMS, user costs, agency costs, MRR costs, deterioration models, etc. were developed in a detailed manner. Similarly, PONTIS has been replaced by AASHTOWare in recent years. Several studies (Bai *et al.*, 2013; Ghodoosi *et al.*, 2018; Cheng, Yang and Frangopol, 2020; Li *et al.*, 2020) have focused on the methodologies and shortcomings of BMSs. Similar research has also been conducted on other assets such as pavement (Lamptey, Labi and Li, 2008), geotechnical assets (Thompson *et al.*, 2016), and sewage systems (Cardoso, Almeida and Santos Silva, 2016).

But, there is currently no open-source and freely accessible platform written by a flexible programming language for IAM.

Chapter 3 An open-source and extensible platform for general infrastructure asset management system

The idea of open collaboration in asset management could unleash new opportunities. The major objective of this chapter is to present an open-source asset management platform that is freely available for further development through open collaboration among practitioners, researchers, and programmers in both academia and industry. This platform is called the "general infrastructure asset management system" (GIAMS). Using bridges as an example asset, this chapter introduces the major blocks of the platform for both PL-IAM and NL-IAM. In addition to a technical overview, guidelines for the future development of these blocks from a programming perspective are also provided. These modular blocks are developed with an object-oriented mindset and the intention of facilitating the extension of GIAMS. Given the nature of the asset management problems from a programming perspective, the builder method has been chosen as the design pattern of GIAMS. The builder method is a design pattern that is mostly used when a final product (e.g., an IAM system) aggregates several other blocks and modules to share various tasks and responsibilities among them (Gamma *et al.*, 1995). The design and development of these blocks are mostly based on previously used methods and algorithms of IAM systems. Currently, GIAMS can provide optimized maintenance, rehabilitation, and reconstruction plans for the life cycle of an asset. It can also yield a prioritized portfolio of assets that need to be maintained or rehabilitated in a network. With minor modifications, it can provide optimized structural health monitoring schedules. The Python code of GIAMS is freely accessible online in a GitHub repository (https://github.com/vd1371/GIAMS) (Asghari and Hsu, 2020), and ready for use and further development. Given the inextensibility and licensed nature of previously designed IAM systems written in less flexible programming languages, the main contribution of this study is

to provide a freely accessible asset management platform for practitioners and researchers focusing on this area. This platform can eventually accelerate studies on asset management globally and could also be used in under-privileged communities to support more informed, methodical, sustainable, and profitable decision making.

The applicability of GIAMS is illustrated through two examples based on the Indiana, US, bridge network derived from the National Bridge Inventory (NBI). Realistic models and data from NBI, IBMS (Sinha *et al.*, 2009), HAZUS (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003), and United States Geological Survey (USGS) (USGS, 2020) were used to analyze these two examples. In the first example, the life cycle cost of one bridge in a 20-year management horizon was optimized by the genetic algorithm (GA) and Monte Carlo simulation. In the second example, a portfolio of suggested MRR plans for all bridges, which is a multichoice multidimensional knapsack problem, was optimized using the IUC heuristic method. While these examples are focused on bridges and bridge networks, GIAMS could be further developed and expanded to be used in other asset management areas such as pavement management.

The rest of this chapter is structured as follows. In Section 3-1, a relatively thorough description of the modules and sub-models which are usually used in asset management systems as developed in the GIAMS structure is provided. The illustrative examples based on the Indiana bridge network are described to show a part of the current capabilities of GIAMS in Section 3-2. The results of the illustrative examples are provided in Section 3-3. Technical aspects of open collaboration, key terminologies, and main processes in open collaboration for software development are briefly summarized in Appendix A. Appendix B provides examples for extending this framework using three cases: 1) development of a new

deterioration model, 2) prioritizing seismic retrofit of structure, and 3) maintenance optimization with structural health monitoring results. The location and role of all classes and objects which will be discussed later in the manuscript can be found in the directory tree in Appendix C. Appendix D provides the depiction of information modeling for the studies examples in this study.

3-1 Modules of the framework

In this section, an overview of the main modules of GIAMS's management framework is provided. These modules and their interrelationship are inspired by previous asset management studies focusing on different assets such as flood defenses, railways, ports, wind energy assets, buildings, and bridges (Hall *et al.*, 2003; Jafari and Valentin, 2017; Khouzani, Golroo and Bagheri, 2017; Torres-Machi *et al.*, 2017; Zhang *et al.*, 2017; Shafiee and Sørensen, 2019). Four major modules shape the current structure of this framework. First is the asset (e.g., bridge) module, which consists of several models to explicitly take into account various characteristics and natural phenomena governing the asset. Second is the network module, which aggregates the assets of a network, delineates the limitations (e.g., budget limitation), and yields the objectives for the following modules. The third is the life cycle analyzer module, which evaluates the costs (user costs and agency costs) and performance of an asset (based on the utility theory) in a life cycle with the help of a simulator. Fourth is the optimizer module, which provides an optimal or a near-optimal MRR plan, both for an asset or a network of assets. Figure 3.1 depicts these modules and the system architecture of GIAMS.

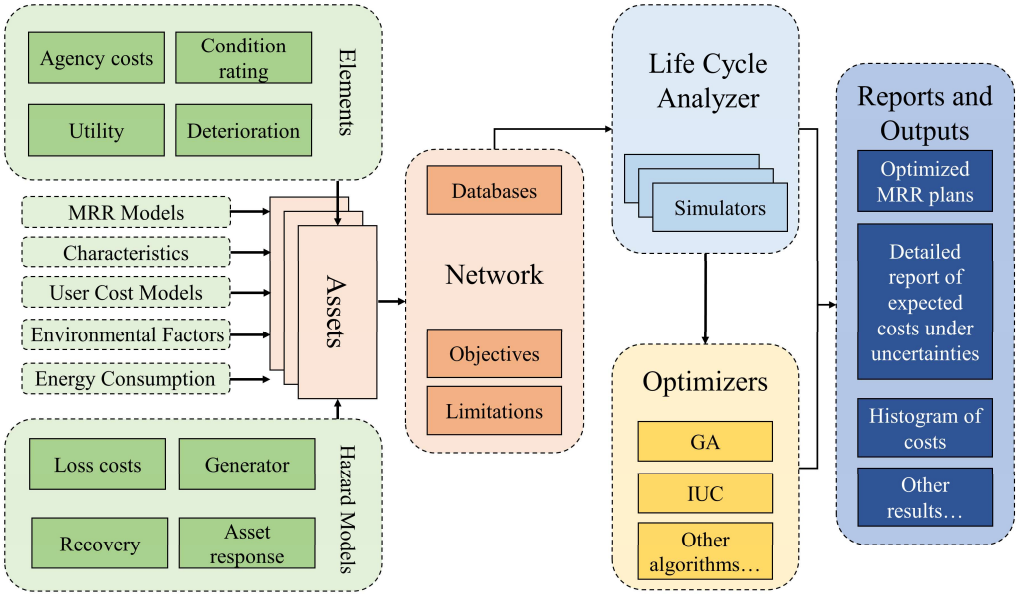


Figure 3.1. The system architecture and logical flow of the modules of the proposed framework

This study employs the notion of object-oriented programming to form different modules, models, and their connection to each other. Object-oriented programming is a well-established concept in software programming that has provided the opportunity for programmers around the world to design and develop products easier and cleaner. Almost all current famous and well-known programs are written with an object-oriented programming mindset. In object-oriented programming, each program consists of several programming objects/classes that connect to each other, inherit from one another, and work together to perform specific tasks. Each object has attributes that demonstrate its specific characteristics and methods that represent its functionality. These objects are usually set aside by each other following standard design patterns. In programming, a design pattern is a solution to a general problem intended to reduce implementation time and increase the efficiency of developing and extending a software program. The design pattern has three major categories (i.e., creational patterns, structural patterns, and behavioral patterns), each of which is suitable for handling a specific problem. The builder method is a subtype of the creational

pattern. This method is usually used when a system of multiple components is independent of how the products are created and composed. It is also used when a family of related product objects should be aggregated to create a final product. Following the software engineering practice, all modules, sub-modules, and models that are currently developed in GIAMS have undergone testing to ensure the correctness of the behavior of each unit.

To further explain the logical flow of Figure 3.1, the relationships between these blocks are explained in reverse order in the following. The ultimate purpose of an asset management system is to optimize the LCC or utility of a network of assets. The optimization is usually conducted on the results of the LCCA (e.g., costs and utility) of a network of assets or one asset. Since different practices and simulation technics might be required in the LCA to get different outcomes, a simulator module should be placed in the LCA block. The simulator module makes random realizations of probabilistic events in the life cycle of an asset. Another main component of the LCA is the network of the assets. Each network of assets has its limitations (e.g., monetary limitations) and objectives (e.g., maintaining the CR of 90% of assets above a certain level). One or some assets could be a part of the network module. Optimization and LCA modules cannot directly receive the limitations and objectives of a network since they are general mathematical and simulation tools for general purposes. Being subject to different hazards in the life cycle, each asset has several elements, characteristics, an MRR plan, and a user cost module. The asset's components are thoroughly discussed in Section 3-1-1. It should be noted that different assets experience different magnitudes of hazards that occur in a network. Similarly, other components of the hazard module such as response, loss costs, and recovery plans are also asset-specific models. Therefore, the hazard module is considered a building block of each asset. Further explanations of these modules can be found in Section 3-1-1-3. Both programming and technical concerns have been

considered in designing the relationships between the blocks and the logical flow of GIAMS. Figure D. 1. also illustrates the information modeling of the illustrative examples in this study. Notably, future developed asset management systems using GIAMS would follow a similar flow of information.

The extension and development of this framework consist of writing new blocks guided by a given pattern, performing analysis, validating the code, and sharing it with others. Most of these blocks are general modules that could be used directly or with minimum modifications for other types of assets. In those limited cases where problem-specific modules are developed, new modules could be easily developed for new problems. Intending to maintain the integrity of future developed modules/codes with an object-oriented programming mindset, a *Base* model is defined for all models. For example, all current and future developed assets must inherit from the *BaseAsset* model. As a result, attributes and methods of future developed objects must use similar terminologies to that of the *Base* models and overwrite them. Similarly, other modules and models should inherit from their corresponding *Base* model in the repository to maintain the integrity of this framework. The idea of inheritance from the *Base* models and using similar terminologies will simplify modification and update of the framework. In other words, the new blocks should imitate the currently developed blocks to perform similar tasks based on similar inputs and return similar outputs. GIAMS' blocks, as well as their inputs and outputs, are described both from technical and programming perspectives in the following sections.

3-1-1 Asset modules

The asset is the principal component of any asset management system. Consisting of several elements, it is affected by several natural or man-made phenomena and incurs costs on the community. In the context of this study and without loss of generality, a bridge, as a transportation lifeline, has been chosen to demonstrate the flow of constructing an asset

management system. The asset module has several characteristics and is built upon sub-models to capture real incidents and phenomena in the modeling procedure. Asset characteristics are attributes that are set in the asset class, while sub-models are instantiated objects assigned to the asset. The characteristics and sub-models employed in this platform are discussed in the following sections.

3-1-1-1 Asset characteristics

Some parameters and characteristics are shared among various types of assets while some only belong to one type of asset. This makes it necessary to develop an independent sub-model to carefully address the needs for analyzing the intended asset of the study. To illustrate, the seismic response model of bridges differs from that of buildings. Each bridge has traffic-related characteristics (e.g., average daily traffic and road class), seismic characteristics (e.g., site class and HAZUS classification), and other characteristics such as the number of spans, ID, and skew angle. From a programming perspective, these types of information are attributes of the assets and should be a part of the Asset module to make the code readable and concise. Similar to the currently developed assets (i.e., bridges and buildings), other assets can be developed and be used. The full characteristics of this module could be found in the repository of GIAMS.

3-1-1-2 Elements, condition rating, and deterioration

Since assets are typically built upon several elements, GIAMS is designed in a way to meet this need. If an asset has only 1 element or only 1 element is being studied, then one element can be defined and used. To further illustrate, bridges have three main sets of elements (AASHTO, 2015), and one or some of these elements have been used in asset management studies previously. For instance, Yang and Frangopol (David Y. Yang and Frangopol, 2020a) merely used the structure element, while Sinha *et al.* (2009) used deck, substructure,

superstructure, and wearing surfaces in their studies. From a programming perspective, elements are developed in a way to be properties of the asset module to access their information easily. These elements deteriorate over time and need to be monitored, maintained, rehabilitated, or reconstructed. These improvement actions incur costs that are taken on by the relevant agencies based on elements' attributes. In GIAMS, condition monitoring, deterioration models, and agency costs are the input objects/models of the element class. An element object is ultimately assigned to the asset module to be used in the next modules (e. g., the LCA module). The components of a sub-model that should be aggregated to create the element are also depicted in Figure 3.1.

Assets' elements deteriorate over time due to constant exposure to natural agents, chemicals, and loading. Several CR schemes have been proposed to monitor the state of an asset in the face of deterioration. These ratings could be continuous (e.g., the continuous space in [0,1] interval) or discrete (good, fair, poor, severe (AASHTO, 2015)). In the case of bridges, discrete CRs have been used in several famous BMSs such as PONTIS (Thompson *et al.*, 1998), BRIDGIT (Hawk and Small, 1998), and IBMS (Sinha *et al.*, 2009). Table 3.1. summarizes the common discrete numbering in the US, China, Korea, and Japan(Jeong *et al.*, 2018).

Table 3.1. Common CR schemes around the world

NBI rating description	**NBI**	**PONTIS**	**China**	**Korea**	**Japan**
As new	9	1	A	A	I
No problems noted	8				
Some minor problems noted	7	2	B	B	II
Structural elements show some minor deterioration	6				
All primary structural elements are sound but may have minor section loss, deterioration, spalling, or scour	5	3	C	C	

Advanced section loss, deterioration, spalling, scour	4				
Loss of sections etc. has affected primary structural components. Local failures are possible. Fatigue cracks in steel or shear cracks in concrete may be present	3	4	D	D	III, IV
Advanced deterioration of primary structural elements. Fatigue cracks in concrete may be present or scour may have removed structural support. Unless closely monitored it may be necessary to close the bridge until corrective action is taken	2				
Major deterioration or loss of section in critical structural components or obvious vertical or horizontal movement affecting structural stability. Bridge is closed to traffic, but corrective action may be put back in light service	1	5	E	E	
Out of service, beyond corrective action	0				

*1: (FHWA (Federal Highway Administration), 2012), 2: (Thompson *et al.*, 1998)

CR class, which inherits from *BaseConditionRating*, is a set of predefined rating models and is used as the input of the element model in the present study. Similar to the parent object, future developed CR modules must contain a "ratings" attribute that contains the CRs of the assets. Although several CR models for bridges (based on NBI, PONTIS, and systems used in other countries) are currently provided in GIAMS, similar models could be later incorporated into the platform. It should be noted that due to technical concerns regarding the implementation of these CRs, it was decided to use a numeric system starting from 0 in increasing order from best to worst condition. For example, 0 instead of 9 and 9 instead of 0 should be used in the NBI rating. Similarly, the Korean rating would turn to the 0-4 rating system.

Deterioration is a stochastic process affected by several factors including but not limited to weather conditions, age, and construction quality. In one of the earliest studies of asset

management systems, Golabi *et al.* (1982) used the Markov chain to model the transition between one state to another. This approach has been adopted in various asset management systems (Hawk and Small, 1998; Thompson *et al.*, 1998; Sinha *et al.*, 2009). In this context, Markov chain models have a limited number of states (i.e., CRs). It is assumed that there is a fixed transition probability P_{ij}, from one state, i, to the next, j. Besides, a state can only change one level at a time, i.e., the state can change from 2 to 3, but not from 2 to 4. Figure 3.2 depicts an example of a Markov chain matrix with 5 states that could be used in asset management.

$$\begin{bmatrix} P_{11} & 1-P_{11} & 0 & 0 & 0 \\ 0 & P_{22} & 1-P_{22} & 0 & 0 \\ 0 & 0 & P_{33} & 1-P_{33} & 0 \\ 0 & 0 & 0 & P_{44} & 1-P_{44} \\ 0 & 0 & 0 & 0 & P_{55} \end{bmatrix}$$

Figure 3.2. A Markov chain matrix with 5 states

Although this phenomenon is stochastic, several studies (Ellingwood, 2005; Sinha *et al.*, 2009; David Y. Yang and Frangopol, 2020a) have proposed empirical methods based on available data to predict the asset's condition in time that could be used in this regard. Regardless of the formula, the deterioration object inherits from the *BaseDeterioration* object. Similarly, deterioration models developed in the future should follow this pattern. The *predict_method* of the deterioration model provides a probabilistic condition of an element after a time step given a previous condition. Its results will be used in the simulator model of the LCA module.

Implementing MRR incurs direct and inevitable costs. Given the type of actions, be it maintenance, rehabilitation, or reconstruction, the responsible agencies are required to provide the financial resources for carrying out MRR plans. Therefore, it is essential to formulate these costs effectively. Linear models, Wiener process (Ross, 2010), Geometric

Brownian motion (Ross, 2010), Cobb-Douglas models (Sinha *et al.*, 2009; Nicholson and Christopher, 2011), and an aggregation of several expert judgments and historical data (Liu *et al.*, 2018) are some examples of methods that could be used to model agency costs. Any agency cost model to be developed in GIAMS in the future must inherit from the *BaseAgencyCost*. The *predict_series* method of the agency cost objects returns the values required for life cycle analysis in the simulator of the life cycle analysis.

Each MRR action has a different utility for the agencies. As the ultimate goal of agencies is to maximize utility and minimize costs at the same time, utility functions need to be properly estimated. This estimation could be performed via surveys and calibration of a function to the survey results via regression (Sinha *et al.*, 2009). Li and Sinha (2004) provided several utility functions for the case of bridges that have been used in the literature (Bai *et al.*, 2013). The utility of elements could be congregated and form the utility of assets with equal or different weights (Bai *et al.*, 2013). For future developments in other problems, the utility object of each element should be inherited from *BaseUtility.* Its *get* method returns the utility of any conducted action based on a *utility_function* method for the element.

3-1-1-3 Hazard models, occurrence models, and response models

The response of assets in the face of probable hazards, as well as the incurred loss costs and post-hazard recovery strategies, should be modeled meticulously to represent various aspects of hazards both before and after the occurrence. The hazard model consists of four main sub-models in this regard, shown in Figure 3.1. While hazards occur in a network, assigning the hazard module to the asset module is advantageous in some ways. First, only one (i.e., Hazard generator) out of these four sub-models could be assigned to the Network module, and other sub-models like asset response, loss costs, and asset recovery are asset-specified modules. In this case, the platform is easily understandable and concise. Second, although

hazards occur in a network, assets experience them in different ways. Therefore, generating the intensity of a hazard, that an asset experiences, would be easier with the current composition of modules.

Hazards such as earthquakes are unpredictable events that could happen at different times with different magnitudes. Despite their unpredictable nature, the distribution and occurrence rate of a given hazard can be estimated based on historical data. Such data enables random sampling of hazards based on the distributions of occurrences and magnitudes (Talebiyan and Mahsuli, 2018). The Poisson point process (PPP) is a method that can be used to model random phenomena (Ross, 2010). The interval between occurrences is modeled by the Poisson process, Eq. (3.1), which gives the probability of n occurrences with a rate of ζ during period T:

$$P(n) = \frac{(\zeta T)^n e^{-\zeta T}}{n!} \quad (3.1)$$

After generating the occurrence times of the hazards, the magnitude of the hazard is sampled based on its magnitude distribution. The distribution and occurrence rate of the hazards are the input parameters of the currently implemented generator class, PPP. Other hazard generator models that must inherit from the *BaseGenerator* object could have different input parameters. In any case, the hazard generator model must consist of a *generate_one_lifecycle* method that will be called during the simulation. This method returns the time and magnitude of a generated sample of hazards on a horizon.

Different assets behave in different ways towards hazards. Response models are required to be designed to reflect these variations. HAZUS – MH2.1 (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003) is a technical manual with a focus on hazard responses of assets. The general approach of this manual is to use a type of probabilistic model called fragility curves, shown in Eq. (

3.2). The purpose of using these curves is to find the probability of exceedance from a certain damage state given a hazard magnitude and the asset characteristics. In the case of bridges, these models are a function of bridge classification, spectral accelerations at 0.3 sec (Sa0.3), 1.0 sec (Sa1.0), and peak ground acceleration.

$$P_{S \geq S_i | IM} = \Phi \left\{ \frac{1}{\beta_{S_i}} \ln \left(\frac{IM}{m_{S_i}} \right) \right\} \quad (3.2)$$

where $P_{S \geq S_i | IM}$ is the probability of exceedance of the state of assets from S_i, Φ is the standard normal cumulative distribution function, IM is the ground motion intensity measure, m_{S_i} is the median value of ground motion intensity with damage state i, and $\beta_i(S_t)$ is the dispersion factor of damage state i. A fragility-based response model and a pre-tuned model for different bridge classifications are currently provided in the proposed platform. Future response models must inherit from the *BaseRespone,* in which its *get* method returns the response of the asset given a specific hazard. The results of the response model are twofold in GIAMS and include the damage state and the mapped condition based on the damage state. These output parameters will be used in the lifecycle analysis model.

The occurrence of a hazard not only disrupts the functionality of an asset or in some cases leads to its collapse, but also can cause fatalities and casualties. These casualties and fatalities differ for different assets. In other words, the fatalities of buildings are different from those of bridges. These values are also provided by the HAZUS – MH2.1 (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003) in a probabilistic manner. Although the manual does not provide bridge casualty data the option to assess losses due to a hazard is provided in GIAMS. Any future designed and developed loss object inheriting from the *BaseLoss* must have a *predict_series* method that returns the loss costs of an asset given a certain damage state in

the management horizon. Similar to other models, the output of this model will be used in the life cycle analysis model.

After the occurrence of a disastrous hazard, affected assets must be recovered to maintain the asset functionality for the community. Agencies and decision-makers should have pre-determined plans for various after-hazard circumstances to evaluate the LCC and conduct optimization. In other words, they should determine the course of actions to be taken after hazards if the asset would be in intact, slight, moderate, extensive, or collapsed damage states. This option has also been provided in this framework to make simulation and analysis closer to reality. Inheriting from the *BaseRecovery,* any recovery object must contain a *get* method that returns an after-hazard recovery plan given a certain CR.

3-1-1-4 MRR models

Maintenance and other recovery actions are attributes of assets in asset management analysis. Therefore, the MRR models are assigned to the asset model to represent the course of actions with specific encodings (e.g., binary and decimal). Typically, four different types of actions are considered in asset management: maintenance, rehabilitation, reconstruction, and do nothing (Hawk and Small, 1998; Thompson *et al.*, 1998; Sinha *et al.*, 2009). Two different types of encoding, binary and numerical, have been used in optimization and life cycle analysis, shown in Table 3.2.

Table 3.2. MRR actions encoding

Action	Abbreviation	Numerical encoding	Binary encoding
Do nothing	DONOT	0	00
Maintenance	MAINT	1	01
Rehabilitation	REHAB	2	10
Reconstruction	RECON	3	11

To implement a vectorized MRR, this framework uses a 3-dimensional array in its computation core. The first dimension corresponds to the asset, the second dimension corresponds to elements, and the third dimension corresponds to the year in which the MRR action should take place. Illustrating the vectorized form of the MRR in this framework, Figure 3.3 shows an example biennial 20-year MRR plan for a network with 2 bridges with 2 elements.

		0-2	2-4	4-6	6-8	8-10	10-12	12-14	14-16	16-18	18-20	
Bridge 1 – Element 1	[[[	0	2	0	0	0	1	0	0	0	0	]
Bridge 1 – Element 2	[	0	1	0	0	0	3	0	0	0	0	]]
Bridge 2 – Element 1	[[	0	0	0	3	0	0	0	0	0	1	]
Bridge 2 – Element 2	[	0	0	0	3	0	0	0	0	0	0	]]]

Figure 3.3. An example of vectorized MRR for 2 bridges with 2 elements

Some asset management studies also focus on retrofit planning (Talebiyan and Mahsuli, 2018) of structure. These areas of study usually focus on binary choices (Do Nothing and Take Action). To meet this need, GIAMS also contains a two-action MRR module for binary choices that could be used for similar purposes. However, future MRR models could be implemented in GIAMS with various substantial changes and revisions in the modules of the platform.

Another noteworthy issue regarding the MRR plans is the effectiveness of actions. Effectiveness refers to the degree that which an MRR action would ameliorate the condition of the asset. For instance, maintenance of an element with major deterioration or section loss would not improve its condition greatly. On the other hand, reconstruction of any element would change its condition to an as-built state. Inspired by their use in PONTIS (Thompson *et al.*, 1998), Markov chain models have been adopted in GIAMS to inform the model regarding the extent of improvement. An example of such a model for rehabilitation is shown in Table 3.3.

Table 3.3. An example of the transition probabilities as a result of MRR actions and deterioration for an asset with 5 CRs

State	Action	1	2	3	4	5
1	DONOT	0.98	0.02	0	0	0
	MAINT	1.00	0	0	0	0
	REHAB	1.00	0	0	0	0
	RECON	1.00	0	0	0	0
2	DONOT	0	0.95	0.03	0	0
	MAINT	0.96	0.04	0	0	0
	REHAB	1.00	0	0	0	0
	RECON	1.00	0	0	0	0
3	DONOT	0	0	0.88	0.12	0
	MAINT	0.63	0.27	0.10	0	0
	REHAB	0.90	0.10	0	0	0
	RECON	1.00	0	0	0	0
4	DONOT	0	0	0	0.87	0.13
	MAINT	0.50	0.15	0.15	0.20	0
	REHAB	0.80	0.15	0.05	0	0
	RECON	1.00	0	0	0	0
5	DONOT	0	0	0	0	1.00
	MAINT	0	0	0	0.02	0.98
	REHAB	0.03	0.10	0.17	0.30	0.40
	RECON	1.00	0	0	0	0

Any further developed effectiveness model must inherit from the *BaseEffectiveness* module and must contain a *get* method. This method receives the condition of the asset and the proposed action as input and returns the condition of the asset as output. This output of the effectiveness models is then used in the life cycle analysis.

3-1-1-5 User cost models

MRR actions usually result in a heavy burden placed on the stakeholding community. This cost, which is usually called user costs, could be many times greater than the agency costs (David Y. Yang and Frangopol, 2020b). For example, if an agency wants to rehabilitate the deck of a bridge, it will typically need to close one lane or the whole bridge. Drivers will then have to reduce their speed because of a reduced number of lanes, or they must choose a detour to get to their destinations. Detours are longer than the route that includes the bridge and have less maximum travel speed. This means thousands of additional hours and gallons of gas consumed because of one bridge rehabilitation. This concept holds for other types of assets. User cost is usually a function of assets' characteristics and their MRR plans. Although user cost models could be assigned to other modules of GIAMS, assigning them to the asset module would make the platform codes concise and clean. Several methods have been described to estimate the user costs (Sinha *et al.*, 2009; TexasDOT, 2020), details of which are beyond the scope of this article. A user costs model based on the Texas Department of transportation is provided in the framework of this study. Future user cost models will have to inherit from the *BaseUserCost* which makes having a *predict_series* method in it necessary. The output of this model will also be used in the life cycle analysis.

3-1-2 The network module, objectives, and limitations

Asset managers and agencies are in charge of maintaining several assets in a network. Developing a network module follows developing asset modules and sub-models. The network module in this framework plays the role of forming a network by aggregating all assets, yielding the network objectives in the optimization process, and imposing the network constraints. The aggregation part has the responsibility of loading the assets from a data file and converting them in a way so that the framework can understand and conduct analysis upon it. An example of the network module has been provided in the repository for further

modifications. Modifying the network module is one of the earliest steps in using this platform for other problems. Inheriting from the *BaseNetwork*, any network object must contain a *load_network* method that loads all the assets and assign them to the network as the *asset* attributes. The network object itself will be assigned to the LCA module for further analysis.

Although limitations and objectives are usually considered attributes of the optimization modules, they are assigned to the network module. GIAMS is mainly designed in this manner to maintain the flexibility of optimization modules as general computational tools rather than providing problem-specific tools. Intending to optimize the MRR plans, the network module also yields objective values, which can be generated in a minimization or maximization form. The objective function of this analysis could be:

$$\begin{aligned} &minimize\ f_1(x) \\ &maximize\ f_2(x) \\ &\dots \\ &minimize\ f_m(x) \end{aligned} \qquad (3.3)$$

In the case of having more than one objective function, which is called multi-objective optimization, a widely used approach is to transform all objective functions into a single objective using a multiattribute utility function method (Bai *et al.*, 2013). Maximizing the final utility function, Eq. (3.4), would then be the new objective.

$$maximize\ \ f(f_1(x), f_2(x), \dots, f_m(x)) \qquad (3.4)$$

The utility of a network could be the sum of the utility of its assets. However, Bai et al. (Bai *et al.*, 2013) proposed the concept of Holism and raised the additivity problem of the non-linear utility functions. Instead, they proposed applying the utility function to the average of the network CR. Both of these methods for calculating the utility of a network are readily provided in GIAMS. These objective values (utilities) are optimized under several constraints

in asset management. These constraints are usually in two forms (Bai *et al.*, 2013): budget constraints and performance constraints. Budget constraints represent the maximum amount of money that agencies are capable of investing in the MRR of the assets and can be formulated as shown in:

$$\sum_{i=1}^{n} X_i C_i \leq B_d \quad (3.5)$$

where X_i is a binary value indicating the presence of project i in a portfolio, C_i is the cost of project i, and B_d is the budget. Similar constraints could be imposed on the performance level of a network. Performance constraints are vital to decision-makers who want to keep their network at an acceptable level of performance. In the case of bridges, for example, the upper bound of performance would be the maximum number of assets in a poor condition and the lower band of performance would be the average of the assets' CR in the network. The performance constraint can be described as:

$$f^L \leq f(x) \leq f^U \quad (3.6)$$

where f^L and f^U are the lower and upper bands of constraints, and $f(x)$ is an objective function.

3-1-3 Life cycle analysis

The life cycle in this framework refers to the investment and analysis horizon of an asset. This decision horizon could vary for different assets of a network (Sinha *et al.*, 2009) or could be a fixed value (Hawk and Small, 1998). The range and period of the asset management horizon could also vary drastically. It could be as low as 20 years (Hawk and Small, 1998) up to several decades (David Y. Yang and Frangopol, 2020b). Decisions are taken periodically based on new information gathered by professional inspectors. In the case of bridges, for example, the condition and other data on bridges are gathered every 24 months (FHWA (Federal Highway Administration), 2012). Example costs that could be placed upon

an agency and community given the condition of a bridge and MRR plans are depicted as a cash flow diagram in Figure 3.4.

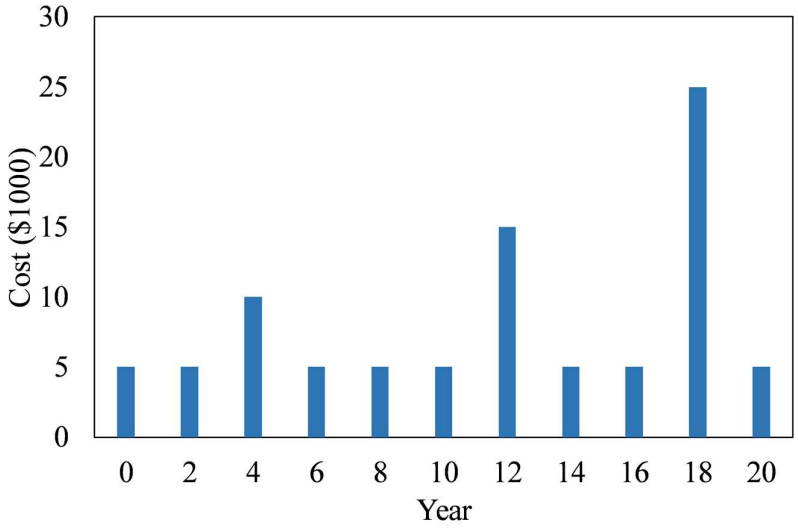


Figure 3.4. Example of cash flow and MRR for an asset

To compare these costs, they can be discounted by Eq. (3.7) to the present time with a discount factor:

$$NPV_{Costs} = \sum_{t}^{N_T} \sum_{j=1}^{N_p} \frac{C_{ij}}{(1+r)^t} \quad (3.7)$$

where C_{ij} is the cost of project j in the t^{th} year, r is the discount factor, N_p is the number of projects, and N_T is the number of decision-making steps in the investment horizon.

The life cycle analysis could be conducted using simple calculations if all models are deterministic (Thompson *et al.*, 1998), or using scenario sampling (Rahimi and Mahsuli, 2019), a variant of Monte Carlo simulation, to get the expected results if uncertainty lies within the models (Liu *et al.*, 2018). In such cases, the life cycle analysis module requires a simulator to create samples of all possible incidents in a life cycle. Figure 3.5 shows an example of all incidents that have happened to an asset in a 20-years horizon.

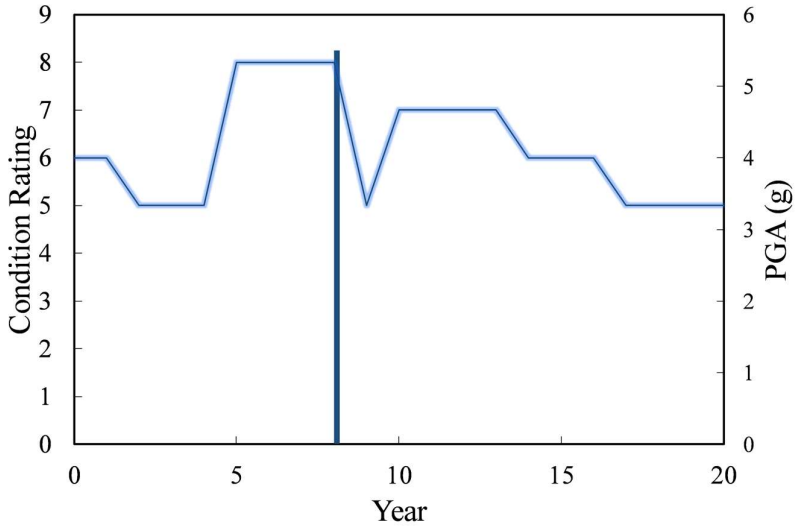


Figure 3.5. An example of life cycle incidents for an asset

In this example scenario, the initial CR of one element of an asset is 6 on the NBI scale. Later, deterioration alters its CR resulting in a CR of 5. Rehabilitation in the 4th year improves its condition, but an earthquake with a PGA of approximately 5.5 during the 8th year results in a CR of 5. Then recovery actions restore its condition to a CR of 7. The element's CR continues to change, affected by deterioration until the end of its management horizon. These events, including sampling hazards, response to hazards, and deterioration, are probabilistic phenomena. An MCS should be used to calculate the expected life cycle results of one MRR strategy under various uncertainties.

The simulator object serves as the computation core of the proposed platform. It generates samples for each asset in the life cycle and returns the analysis results based on the MRR of the asset. Although most common incidents and phenomena governing the assets are currently incorporated into GIAMS, other factors could be added through further development. The simulator object inherits from the *BaseSimulator* and has a *get_one_instance* method. The outputs of this method, i.e. the user costs, agency costs, and

utilities, will be used in the LCA module. The LCA class receives a network and a simulator as input objects and performs a life cycle analysis using the simulator on the network. The results of the LCA usually fall into three main categories: 1) user costs, 2) agency costs, and 3) utilities of the actions. However, the option to get other user-defined types of results from the simulator is also provided in the GIAMS repository. These results, be it deterministic or probabilistic, net present value, or stepwise, are passed on to the optimization module to determine optimal plans.

3-1-4 Optimization modules: project-level and network-level

In this section, the concept and theory of two optimization algorithms that could be suitable for the PL-IAM and NL-IAM are briefly discussed.

3-1-4-1 Project-level optimization and analysis

The purpose of PL-IAM is to inform decision-making about prospective MRR actions concerning the elements of an asset in a life cycle or a management horizon in advance. Hazards, deterioration patterns, as well as different costs, are probabilistic phenomena that could occur in the life cycle. Therefore, an optimized MRR strategy should be selected given various uncertainties in the management horizon. This binary decision-making process could be formulated in a variety of forms. Utility maximization with respect to budgetary limitations and costs minimization with respect to performance limitations are examples of such forms. From another perspective, the process could be approached as a single objective optimization (SOO) or a MOO problem. Given the complexity of MOO problems, a multi-attribute utility function method could be used as an alternative (Bai *et al.*, 2013). This method consists of converting various objectives into dimensionless utilities and then combining them into a single utility. As a result, the original MOO problem will turn into an SOO. All in all, an example of a general form of utility maximization could be:

$$maximize \quad E\left[\sum_{\mathrm{t}=0}^{N_T}\sum_{p\in \mathbf{p}_t} U_{pt}X_{pt}\right] \tag{3.8}$$

With respect to:

$$\sum_{p\in P_t} C_{A_{pt}}X_{pt} \le B_{d_t}$$

$$\sum_{p\in P_t} CR_{pt}X_{pt} \le CR_{min}$$

$$\sum_{p\in P_t} X_{pt} = 1, \qquad for\ each\ year\ \mathrm{t}$$

$$X_{pt} \in \{0,1\}$$

where U_{pt}, $C_{A_{pt}}$, CR_{pt} are utility, agency cost, and CR after the project p is conducted on the t^{th} year, X_{pt} is a binary value if project p is selected on the t^{th} year, CR_{min} is the minimum of the asset (or element) condition, and B_{d_t} is the maximum agency budget on the t^{th} year. Given the complexity of this optimization formula, heuristic algorithms are recommended (David Y. Yang and Frangopol, 2020a) to solve the optimization problem.

The genetic algorithm (GA) (Whitley, 1994) is a heuristic search method that is used to find the maximum or minimum of an objective function. Although GA is computationally expensive and perhaps impractical for big networks, its merits make it desirable for project-level life cycle optimization of assets (David Y. Yang and Frangopol, 2020a). First, it can provide near-optimal MRR strategies with the highest expected utility, though it does not guarantee identifying the optimal solution. Second, GA can be easily understood, implemented, and used by practitioners and researchers. Finally, GA is flexible for use in optimization problems with a variety of forms. Consequently, it could satisfy three out of four conditions, namely accuracy, computation time, robustness, and simplicity, of an appropriate optimization method proposed by Patidar *et al.* (2011). Presumably, due to these reasons, GA

has been the most popular optimization algorithm in the asset management literature as of the late 2020s (Chen and Bai, 2019). It should be highlighted that GA is susceptible to converging to local optima and perhaps not the best optimization algorithm for this purpose. That said, the purpose of this study is to develop a general tool for other researchers to implement their models on it in a more efficient way. In other words, outperforming the previously developed optimization algorithms is not in the scope of this study and is kept for future research.

Inspired by natural evolution, this algorithm seeks a solution in several evolving generations of individuals until the search algorithm meets certain criteria. The common steps of GA optimization are provided in Figure 3.6.

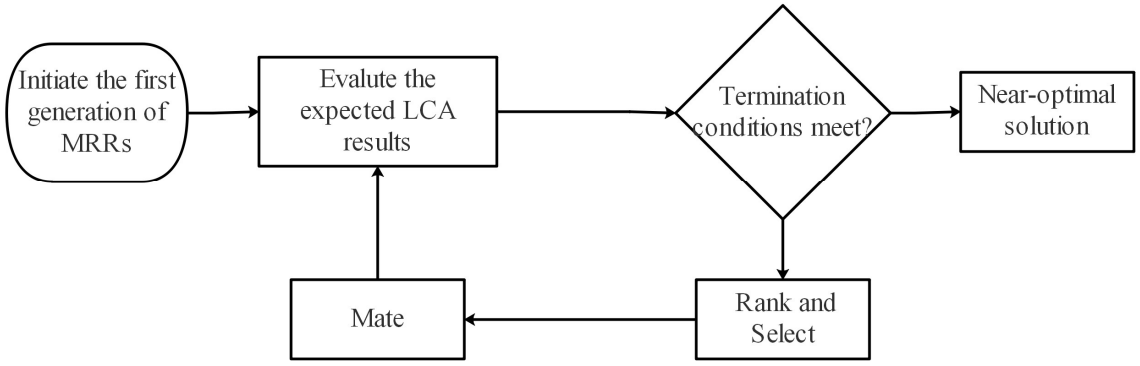


Figure 3.6**.** Genetic algorithm flowchart

Individuals contain a chromosome that represents a point in the search space. In the context of asset management, the binary representation of MRR plans is its chromosome. To illustrate, Figure 3.7 shows the binary representation of the MRR plan in this framework.

	0-2	2-4	4-6	6-8	8-10	10-12	12-14	14-16	16-18	18-20
Bridge 1 – Element 1	[[[0 0	1 0	0 0	0 0	0 0	0 1	0 0	0 0	0 0	0 0]
Bridge 1 – Element 2	[0 0	0 1	0 0	0 0	0 0	1 1	0 0	0 0	0 0	0 0]]
Bridge 2 – Element 1	[[0 0	0 0	0 0	1 1	0 0	0 0	0 0	0 0	0 0	1]
Bridge 2 – Element 2	[0 0	0 0	0 0	1 1	0 0	0 0	0 0	0 0	0 0	0]]]

Figure 3.7. The binary representation of an MRR plan

At first, these chromosomes are initiated to breed the first generation. The performance of individuals indicates their chance of passing their genes to the next generation. The objective function of the optimization problem (e.g., Eq. (3.10)) is used to represent this performance. The selection and mating steps follow the performance evaluation step. Intuitively and similar to natural selection, stronger genes are passed on to the next generation and weaker ones are eliminated. Among several selection methods such as random selection and roulette wheel, the rank-based selection was chosen because of its simple adaptability to both minimization and maximization problems. In rank-based selection, individuals are sorted based on their objective function value and the optimization type. Then they are selected with the probability of:

$$P(j) = \frac{\Gamma_j}{\sum_{i=1}^{n} i} \qquad (3.9)$$

where $P(j)$ is the probability of selection of individual j, Γ_j is the rank of individual j, and n is the number of individuals. After selecting two parents, they will mate and create two new offspring. The mating consists of two phases, namely crossover and mutation, shown in Figure 3.8.

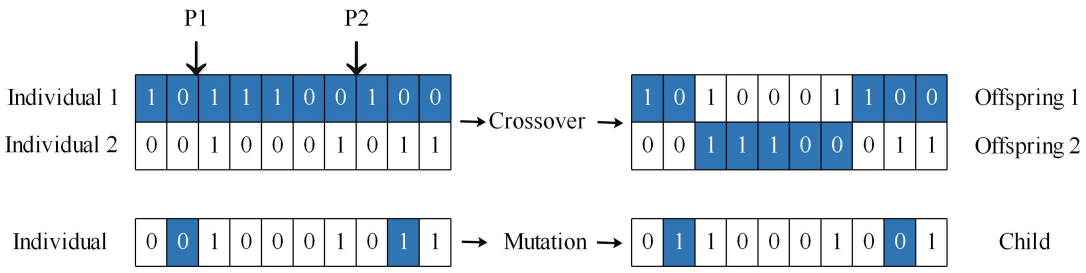


Figure 3.8. Crossover and bit-flip mutation

Some of the top solutions are directly transferred to the next generation to avoid losing the optimal solution during the optimization. They are called the elites of each generation. Some of the hyperparameters of the GA are crossover strategies, crossover probability, mutation

strategies, mutation probability, number of elites, the population size in each generation, number of generations, number of elites, and termination conditions. For details on these hyperparameters, please refer to (Grefenstette, 1986; Witt, 2013; Chicano *et al.*, 2015; Edward Best, 2016).

3-1-4-2 Network-level optimization and analysis

The ultimate purpose of NL-IAM is to allocate the available and limited budget to certain decisions and tasks to improve the overall conditions of an asset or a network, reduce user costs, and sustain the community. The budget allocation for different projects of a portfolio is also a binary choice. This problem is defined as a multichoice multidimensional knapsack problem (MCMDKP) (Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011). Multichoice in this formulation means that exactly one choice, including do-nothing, must be selected for each project. Multidimensional refers to different budget or performance constraints. Although this problem could be formulated in many ways, a general example is provided:

$$maximize \sum_{\mathrm{i}=1}^{n} \sum_{p \in P_i} U_{pi} X_{pi} \qquad (3.10)$$

With respect to:

$$\sum_{\mathrm{i}=1}^{n} \sum_{p \in \mathbf{P}_i} C_{A_{pi}} X_{pi} \leq B_d$$

$$\frac{1}{n}\left(\sum_{\mathrm{i}=1}^{n} \sum_{p \in P_i} CR_{pi} X_{pi} \leq CR_{min} \right)$$

$$\sum_{p \in P_i} X_{pi} = 1, \qquad for\ each\ \mathrm{i}$$

$$X_{pi} \in \{0, 1\}$$

where U_{pi}, $C_{A_{pi}}$, CR_{pi} are utility, agency cost, and CR after the project p is selected for bridge i, respectively, X_{pi} is a binary value if project p is selected for bridge i, CR_{min} is the

minimum of the average network condition, and B_d is the maximum agency budget. Exact methods could yield a solution to an MCMDKP problem. Nevertheless, the computation time for these polynomial time-hard optimization problems grows exponentially with the number of decision parameters. Acknowledging the undesirability of high computational costs (Frangopol, 2011; Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011; Kim and Frangopol, 2018; David Y. Yang and Frangopol, 2020a), a heuristics method could provide near-optimal solutions more quickly, though at the expense of losing the exact solution.

IUC algorithm is a heuristic algorithm that could be used to approximate the result of the knapsack optimization, Eq. (3.10). Another version of IUC, incremental benefit-cost ratio (IBC) has been used in previous asset management systems (Thompson *et al.*, 1998; Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011) for the same purpose. The IUC optimization algorithm is appealing for several reasons. First, it is a greedy heuristic algorithm that yields a near-optimal solution benefiting the project with the highest IUC (Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011). Second, the computation time of IUC optimization is considerably less than that of deterministic methods or some other heuristic methods (e.g., Lagrangian relaxation). Third, it is simple and understandable by prospective users. Finally, it is easily adaptable to variations in the optimization problem. Further discussion on methods for asset management can be found in (Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011).

Within the IUC heuristic optimization algorithm, the potential projects are sorted based on their utility cost ratio, Eq. (3.11) in descending order:

$$UC\left(p_j\right) = \frac{U_{pj}}{C_{p_j}} \qquad (3.11)$$

where U_{pj} is the utility of the project p_j at the cost of C_{p_j}. Then the available budget is allocated to the remaining projects with the highest IUC. This allocation is continued until the agency's budget is exhausted (Algorithm 3.1).

Algorithm 3.1. IUC heuristic

1: $\mathbf{P_{IUC}} = \{\}$ A holder for selected projects

2: $\mathbf{Q_{IUC}}$ = {All projects sorted based on IUC in decreasing order}

3: **for each** p_j **in** $\mathbf{Q_{IUC}}$:

4: **if** $C(\mathbf{P_{IUC}}) + C(p_j) > B_d$:

5: **break**

6: **else**:

7: $\mathbf{P_{IUC}} = \mathbf{P_{IUC}} \cup \{p_j\}$

8: **return** $\mathbf{P_{IUC}}$

Note: $\mathbf{P_{IUC}}$ is the selected portfolio of projects, $\mathbf{Q_{IUC}}$ is the set of all projects sorted based on IUC, and C is the cost of the projects.

3-2 Illustrative examples

Since the 1980s, several BMSs, updates, and research methods have been proposed for the Indiana bridge network (Sinha *et al.*, 2009; Bai *et al.*, 2013). The bridge network in Indiana consists of more than 4,600 state bridges (Bai *et al.*, 2013), with their related information such as structural types, average daily traffic, and CR stored in the national bridge inventory of the US. For illustration, three main components of bridges, namely deck, superstructure, and substructure, were used for life cycle optimization and network budget allocation. In addition, agency costs, user costs, utilities, and deterioration models for each bridge were adopted from IBMS (Sinha *et al.*, 2009) and the Texas Department of Transportation (TexasDOT, 2020). Seismic characteristics such as site class, HAZUS classification, response models, and loss models were derived from the HAZUS manual (FEMA-NIBS: Federal

Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003). Earthquake history data for the surrounding region were also collected from the USGS earthquake database (USGS, 2020). Other models such as effectiveness models and recovery models were inspired by previous BMSs (e.g., BRIDGIT (Hawk and Small, 1998)) or rationally assumed (i.e., the CR of the asset after recovery actions will be similar to that of NBI rating 8). These assumptions and the models can be readily found in the repository of GIAMS (Asghari and Hsu, 2020). Given that a number of previously published articles in the asset management area by the late 2020s were at the project-level (Chen and Bai, 2019), GIAMS was designed so that it could search for near-optimal life cycle MRR strategies by GA. However, the majority of previous studies have focused on network-level optimization (Chen and Bai, 2019). Therefore, IUC has been developed in GIAMS to meet this need across different asset management areas.

Using the bridge network of Indiana, two illustrative examples (one project-level and one network-level) are provided to demonstrate the applicability of the current components of GIAMS. Being derived by GA and Monte Carlo simulation, the first example ("Example1" in the GIAMS repository) is dedicated to the optimization of the utility over a fraction of all costs of one asset. The limitations of this optimization are the agency's annual budget in the management horizon and a maximum of the net present value of all costs. In addition, one element cannot experience two consecutive MRR actions, be replaced more than twice, be rehabilitated more than 4 times, or be maintained more than 6 times in the management horizon. The second example is dedicated to network-level optimization of Indiana's bridge upkeep using IUC. The pool of candidate projects for network-level optimization could be populated according to expert judgments and decision trees (e.g., DTREE (Sinha *et al.*, 2009)). However, a naïve approach has been currently adopted in GIAMS. In this approach, the projects with the highest utility/cost (U/C) ratios out of all possible combinations of MRR

actions for all bridge elements at the decision-making time are considered potential projects to be carried out for that bridge.

3-3 Results and summary

The results of the two illustrative examples are briefly summarized in this section.

3-3-1 Example 1: Life cycle optimization of one asset

The first example focuses on finding an optimized MRR plan for the bridge with structure number 10 in the NBI inventory. The purpose of this optimization is to maximize the utility of the MRR actions over a fraction of all costs in a 20-year management horizon under budget limitations. The optimization is formulated as follows:

$$maximize \quad E\left[\sum_{t=0}^{N_T}\sum_{p\,\epsilon P_t}(U_{pt}X_{pt}/C_{pt}^{0.2}).e^{-rt}\right] \qquad (3.12)$$

where U_{pt} and C_{pt} are utility and all costs (i.e., agency cost + user costs) after the project p is conducted on the t^{th} year, X_{pt} is a binary value if project p is selected on the t^{th} year and r is the discount rate. The discount rate was assumed to be 3% in this study. Table 3.4 provides a summary of the hyperparameters used in this example.

Table 3.4. Summary of GA hyperparameters

Hyperparameter	Value	Hyperparameter	Value
Optimization type	Maximize	Crossover probability	0.75
Population size	100	Crossover method	Two-point
Number of generations	200	Mutation probability	0.03
Number of elites	5	Mutation method	Bit-flip
Selection method	Rank	Number of simulations for each MRR	2000

Random initialization and preference initialization could be used for this first-generation initiation. However, preference initialization has a higher convergence speed (Xuan *et al.*, 2011) and is used in this example. In this approach, the probability of each bit of the MRR

binary array being 1 could be 0.1, 0.2, 0.3, 0.4, or 0.5. This analysis was conducted by an Intel® CORE™ i7-8700T, 2.40 GHz, 12 computational cores, and 8GB RAM in parallel in approximately 492 minutes. The results of the optimization, (i.e., the utility versus generation number) are depicted in Figure 3.9.

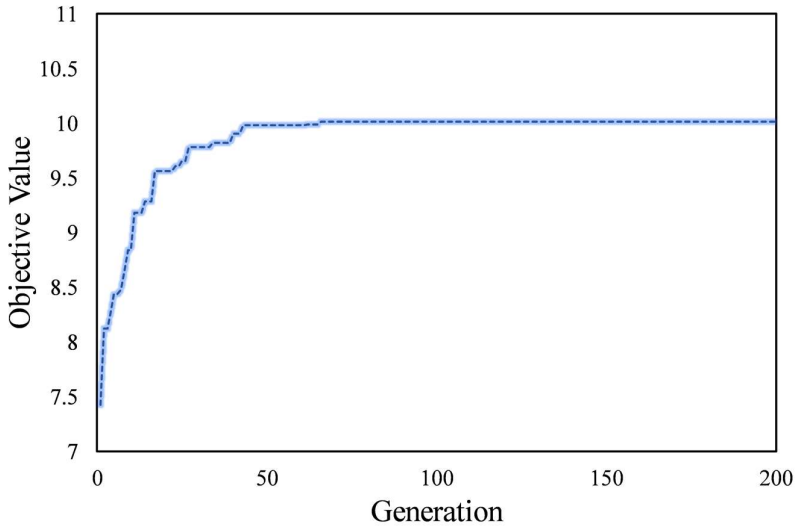


Figure 3.9. The utility of the asset in generations

The results of the optimization suggest a strategy regarding the MRR of the studied bridge, which is depicted in Figure 3.10. The description of the numerical encoding can be found in Table 3.2.

		0-2	2-4	4-6	6-8	8-10	10-12	12-14	14-16	16-18	18-20	
Deck	[[[	3	0	1	0	0	0	1	0	0	1	]
Superstructure	[	2	0	1	0	0	0	0	0	0	1	]]
Substructure	[[	2	0	1	0	0	0	0	0	0	1	]]]

Figure 3.10. The MRR strategy in the management horizon of the example bridge

Given the underlying uncertainties in GA optimization, it is sometimes necessary to validate its results. To this end, different options are provided in GIAMS:

1- Brute force algorithm: Given the complexity of the designed problem and access to sufficient computational resources, GIAMS can enumerate and analyze all possible

combinations of MRR plans. To reduce the computation time, the developed brute force algorithm takes advantage of parallel computing (Yang, Hsieh and Kung, 2012) being able to analyze in parallel on any arbitrary number of computational cores.

2- Benchmarking functions: The performance of a developed GA can be evaluated on some benchmarking functions (Digalakis and Margaritis, 2001). Although several benchmarking functions (e.g., De Jong's function, Rosenbrock's valley function, and Schwefel's function) are currently implemented for validation, the design of GIAMS provides the opportunity for adopting any other benchmarking functions to validate the results of the optimization.

3- Optimizing several rounds: Another approach to validate the results of the GA is to conduct the optimization several times to ensure no convergence to a sub-optimal point (Chen, Chen and Jiang, 2016). This option is currently provided in GIAMS to automatically perform the optimization for a user-defines number of rounds.

4- Other heuristic algorithms: Although the GA is the proposed optimization algorithm for PL-IAM and has been the most used optimization algorithm in the asset management literature (Chen and Bai, 2019), other heuristic algorithms (i.e., PSO and hill-climbing) are designed and developed in GIAMS for further research. Nevertheless, the comparison of the performance of these optimization algorithms with the GA is not within the scope of this study.

3-3-2 Example 2: Network-level project selection

IUC heuristics analysis was conducted by the same computational platform used in the previous example in 8.4 minutes. The output of this analysis is a ranked portfolio of possible projects for bridges in the network at the time of analysis. Table 3.5 presents the top 20 bridges with the highest U/C ratio and their chosen MRR strategies.

Table 3.5. Top 20 bridges with the highest U/C ratio and corresponding MRR actions

Rank	Bridge ID	Deck	Superstructure	Substructure
1	18841	DONOT	DONOT	RECON
2	80132	DONOT	MAINT	MAINT
3	80362	DONOT	MAINT	MAINT
4	80126	DONOT	MAINT	MAINT
5	43020	DONOT	MAINT	MAINT
6	75240	DONOT	MAINT	MAINT
7	37410	DONOT	MAINT	MAINT
8	70520	DONOT	MAINT	MAINT
9	6002	DONOT	DONOT	MAINT
10	31130	MAINT	MAINT	MAINT
11	75200	DONOT	MAINT	MAINT
12	80294	DONOT	MAINT	DONOT
13	33165	DONOT	MAINT	MAINT
14	26570	DONOT	DONOT	MAINT
15	70250	DONOT	MAINT	MAINT
16	76270	DONOT	MAINT	MAINT
17	37300	DONOT	MAINT	MAINT
18	25510	MAINT	MAINT	MAINT
19	80348	DONOT	DONOT	MAINT
20	6003	DONOT	DONOT	MAINT

Chapter 4 Expediting life cycle costs analysis of assets under multiple uncertainties by deep neural networks

To address the limitations discussed in Chapter 1 and Chapter 2, this study puts forth a methodology to estimate LCCA results (e.g., costs and utilities) rather than regular sampling at the time of planning. The proposed methodology enables decision-makers and asset managers to reduce the computation time of complex LCCA of assets by abiding by an acceptable overhead computation time for training a deep neural network model. PL-IAM studies have used ML techniques in the components of LCCA rather than attempting to estimate the LCCA results. Therefore, the primary contribution of this study to the body of knowledge is proposing a methodology to reduce the computation time required for the LCCA by providing a clear set of procedures for synthesizing data and training a deep neural network to estimate the LCCA results. As a result, the MRR optimization of assets could be reached in a far shorter time frame, enabling MRR optimization of each asset in a network without compromising on the complex models and inherent uncertainties in the problem. Although bridges were used as an example asset, this methodology could eventually be applied to other types of assets. The source code of the LCCA framework of this study is available online in the GIAMS GitHub repository (https://github.com/vd1371/GIAMS) (Asghari and Hsu, 2020). GIAMS is an open-source general IAM system that has been previously developed by authors and is freely accessible online. The ML modeling was conducted in the Python programming environment by Tensorflow (Abadi *et al.*, 2016), Keras (Chollet, 2015), and Sci-kit learn (Pedregosa *et al.*, 2011). The codes for training the neural networks are also fully available online (https://github.com/vd1371/XProject).

The remainder of this chapter is structured as follows. First, the main parts of the proposed framework are explained in the methodology section. Then, the key aspects of

project-level bridge management, its components, and the LCCA module in this study are discussed, followed by a case study drawing upon data from the US NBI to illustrate the capabilities of the proposed methodology. Finally, the results and further discussion of them are provided, followed by a summary of key conclusions. The result of this chapter reveals significant improvement in the computation time of complex LCCA with negligible prediction errors.

4-1 Methodology

An overview of the proposed methodology is presented in this section. This methodology primarily consists of three parts: 1) the LCCA module, 2) data synthesizing, and 3) ML model training. A high-level flowchart of the proposed methodology is shown in Figure 4.1, followed by a brief discussion of each of the procedures and sub-procedures.

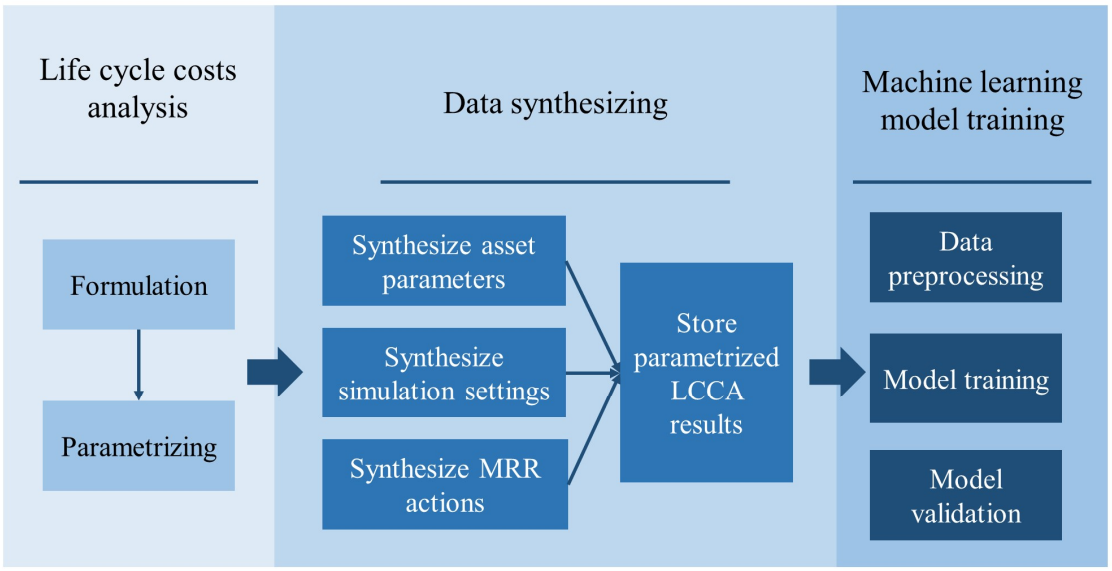


Figure 4.1. Flowchart of the methodology for estimating the LCCA results

4-2 Life cycle costs analysis

The LCCA of an asset refers to the process of evaluating various costs such as construction and maintenance incurred by the asset during its life cycle (investment horizon). When multiple sources of uncertainties and stochastic phenomena exist, MCS could be used to simulate all incidents, their consequences, and their corresponding costs. To briefly explain this approach, the first step is that the CR of assets, which is affected by deterioration, MRR activities, hazards, and post-hazard recovery actions, is simulated in a life cycle. In the next step, agency costs and stakeholders' utilities due to MRR/recovery actions are calculated. Finally, user costs due to transportation delays, excessive fuel consumption, loss of lives, injuries, etc. are evaluated. Using statistical and probabilistic methods, quantitative representations of the simulation results are generated for further analysis and evaluation. These representations could be in the form of a simple average of user costs, agency costs, and utilities (Chen *et al.*, 2015; Frangopol, Dong and Sabatino, 2017):

$$C_U = \frac{1}{N_s}\sum_{n=1}^{N_s}\left(\sum_{t=0}^{T}\left(\sum_{i}\left(X_{it}C_{T_{it}}\right) + C_{L_t|IM}\right)/(1+r)^t\right) \quad (4.1)$$

$$C_A = \frac{1}{N_s}\sum_{n=1}^{N_s}\left(\sum_{t=0}^{T}\left(\sum_{i}\left(X_{it}C_{M_{it}}\right) + C_{R_t|IM}\right)/(1+r)^t\right) \quad (4.2)$$

$$U = \frac{1}{N_s}\sum_{n=1}^{N_s}\left(\sum_{t=0}^{T}\left(\sum_{i}\left(X_{it}U_{\mathrm{it}}\right)\right)/(1+r)^t\right) \quad (4.3)$$

where C_U, C_A and U are expected user costs, agency costs, and utilities, X_{it} is a binary parameter for action i at t, C_T is MRR user costs, C_M is maintenance costs, $C_{L|IM}$ and $C_{R|IM}$ are loss costs and recovery costs given a certain hazard with an intensity measure of IM, U is the utility of actions, r is the discount rate, T is the investment horizon, and N_s is the number of simulations.

4-2-1 Parametrizing LCCA

Asset parameters, MRR actions, MCS parameters, and LCCA results can be converted into a vectorized form. The initial condition of elements, length, width, and degradation rates are some of the assets' parameters. MRR actions are usually a vector of binaries in the management horizon. Simulation parameters could comprise parameters such as inflation rate, hazard occurrence probability and magnitude, and management horizon. These parameters, in their general form, that are incorporated in different models inside an LCCA computational core can be fed to an ML model for estimating LCCA results (Figure 4.2)

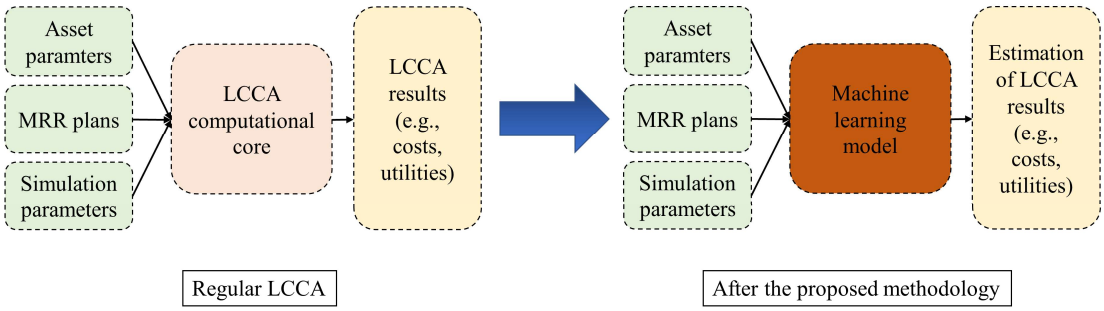


Figure 4.2. The abstract idea of replacing the LCCA computational core with an ML model

4-2-2 Sampling LCCA parameters and results

A large number of samples are required to properly train an ML model. In this context, a large number of $\mathbf{A_p}$, $\mathbf{M}$, $\mathbf{S_{MCS}}$, and $\mathbf{R_{LCCA}}$ (LCCA results) vectors are required to ensure ML models can cover and predict all points of feature space. The $\mathbf{M}$ (MRR actions) and $\mathbf{S_{MCS}}$ (MCS parameters) could be sampled by selecting different actions for MRR plans and different approaches toward simulation. However, the variety of $\mathbf{A_p}$ (asset parameters) is not enough due to the limited number of real assets. Synthesized assets with imaginary parameters based on real assets could be fabricated for training the ML model. This process mostly resembles the data augmentation technique in computer science problems (Redmon *et al.*, 2016) where collecting more data is expensive or impossible. Since the number of

infrastructure assets is limited, synthesizing fabricated assets is an appropriate method to generate a sufficient number of data samples for training an ML model. The number of required samples for having reliable predictions is affected by the feature space size and complexity of the problem. In the present case, millions of assets with different MRR actions and MCS settings would be required to train an ML model with acceptable prediction errors.

4-2-3 Estimating LCCA results

Results of the LCCA computational core can be estimated by an ML model if enough LCCA samples for different bridges and MRR actions are available. The abstract idea of this methodology is depicted in Figure 4.2. Within this process, all or a subset of parameters could be used for ML training purposes. The subsets of parameters include some variables given the experts' and practitioners' requirements. For example, the inflation rate could be considered a constant in one study and a variable in another. The constants should be omitted to avoid increasing the dimension of the problem without adding information to the dataset for training ML models. Finally, estimation performance and the accuracy of results of a trained ML model can be validated by statistical measures such as correlation coefficient and common prediction metrics such as mean absolute percentage error. Depending on problem complexity, sample size, and feature space size, different ML models are subject to strengths and limitations and provide different levels of performance and accuracy.

4-2-3-1 Applicability of different machine learning models

The DNN model is an appropriate choice for estimating the results of LCCA because of three main reasons. First, DNN models have been characterized to be universal approximators that can capture any degree of non-linearity. LCCA of assets with stochastic and non-linear models as well as their results are inherently complex and highly non-linear. A candidate ML model must be able to be trained accurately on this type of dataset. Therefore, linear-based

models such as simple linear regression, Lasso, and ridge would not yield satisfactory predictions. Second, DNN models are updatable. This means that DNN models can be updated with continuing training with the addition of new observations. Since sampling and training on a large dataset might take numerous steps, it would be time-consuming to start training from scratch after receiving new observations. Therefore, DT-based models (e.g., RF, boosting algorithms), k-nearest neighborhood, and support vector machine regression algorithms would be inefficient. Third, DNN training time on big datasets is relatively shorter than other algorithms given recent advances in data science programming libraries/packages. Using graphical processor units (GPU) computational power, for example, TensorFlow (Abadi *et al.*, 2016) can train complex DNN models on regular computers in a relatively short amount of time. The need for a feasible computation time during training sessions renders the support vector machines model unsuitable for this methodology.

4-2-3-2 Deep neural networks: A brief overview

The DNN model is an algorithm widely used in both academic literature and industrial problems. Layers of nodes and neurons interconnected with non-linear activation functions establish a non-linear relationship between the input parameters (independent variables) and target parameters (dependent variables):

$$\mathbf{l}^{i} = \phi(\mathbf{W}^{i}\mathbf{l}^{i-1} + \mathbf{b}^{\mathbf{i}}) \quad (4.4)$$

Where ϕ is the activation function of each layer, $\mathbf{l}^{\mathbf{i}}$ and $\mathbf{b}^{\mathbf{i}}$ are the vectorized results and bias vector of layer i, and $\mathbf{W}$ is the vectorized nodes' weight. Notably, $\mathbf{l}^{\mathbf{0}}$ and $\mathbf{l}^{\mathbf{n}}$ refer to the input vector and target value in a DNN structure with n layers.

A variant of gradient descent algorithms (e.g., RMSProp, Adam) can be used to optimize the weights and biases to maximize the similarity between the predicted and actual target values. Table 4.1 summarizes some of the most common cost functions such as the mean of

squared error (MSE), mean of absolute errors (MAE) or mean of absolute percentage error (MAPE). DNN models have several other hyperparameters (e.g., number of hidden nodes and layers, activations functions, and optimizer) that must be tuned before training. Although hyperparameters tuning is a craft of experience, guidelines have been proposed to optimize this process (Ng, 2018).

Table 4.1. Cost functions

Cost function	Formula
Mean of squared error	$\frac{1}{n}\sum_{i}^{n}(\hat{y}_i - y_i)^2$
Mean of absolute error	$\frac{1}{n}\sum_{i}^{n}(\lvert\hat{y}_i - y_i\rvert)$
Mean of absolute percent error	$\frac{1}{n}\sum_{i}^{n}\left\lvert\frac{\hat{y}_i - y_i}{y_i}\right\rvert$

4-3 Case study: LCC in BMSs

In this section, an illustrative example of the proposed methodology using LCCA in BMSs is provided. Bridges are one of the most important infrastructure assets of a community and have been the focus of many studies by the end of the 2020s (Chen and Bai, 2019). GIAMS, the open-source and freely accessible general infrastructure asset management platform developed in Chapter 3 (Asghari and Hsu, 2020), is used to evaluate the results of the life cycle analysis of bridges in this example. In this section, first, a brief overview of project-level BMSs is provided. Then, details of parametrizing and sampling LCCA results in this example are provided followed by further details of DNN training.

4-3-1 LCCA of bridges in project-level management

Project-level BMSs aim to find the optimal set of actions in the life cycle of a bridge given a limited budget and other constraints (FHWA (Federal Highway Administration), 2012).

Depending on the type of study and problem, deterministic optimization methods such as linear programming (Thompson *et al.*, 1998) or heuristic optimization methods such as GA (Kim and Frangopol, 2018) could be used to minimize the costs and maximize the utilities.

4-3-1-1 Condition rating and monitoring

Bridge elements such as deck, superstructure, and substructure deteriorate over time due to various reasons such as traffic loads and environmental stresses. The condition of these elements should be inspected periodically for further analysis. For example, the bridge data in the US is collected every 24 months and stored in the NBI (FHWA (Federal Highway Administration), 2012). The CR system of bridge elements varies in different BMSs. For example, the NBI uses a discrete CR from 0 to 9, which is summarized in Table 4.2. In addition, HAZUS damage states are mapped to the NBI CR and shown in this table.

Table 4.2. The CR system of NBI

Code	**HAZUS Damage State**	**Description**
9	ds_1	Excellent condition
8		Very good condition – no problems noted
7	ds_2	Good condition – some minor problems
6	ds_3	Satisfactory condition – structural elements show some minor deterioration
5		Fair condition - all primary structural elements are sound but may have minor section loss, deterioration, spalling, or scour
4	ds_4	Poor condition – advanced section loss, deterioration, spalling, or scour
3	ds_5	Serious condition – loss of section, deterioration, spalling, or scour have seriously affected primary structural components. Local failures are possible. Fatigue cracks in steel or shear cracks in concrete may be present
2		Critical condition – advanced deterioration of primary structural elements. Fatigue cracks in steel or shear cracks in concrete may be present or scour may have

	removed substructure support. Unless closely monitored, it may be necessary to close the bridge until corrective action is taken.
1	Imminent failure condition – major deterioration or section loss present in critical structural components or obvious vertical or horizontal movement affecting structure stability. Bridge is closed to traffic, but corrective action may be put back in light service.
0	Failed condition – out of service – beyond corrective actions

4-3-1-2 Markovian deterioration

Deterioration is the first and main source of uncertainty that affects the condition of bridges and the outcomes of LCCA. The first-order Markovian process is a common method for modeling the probabilistic phenomenon of deterioration in infrastructure management when the CRs are discrete (Thompson *et al.*, 1998; Sinha *et al.*, 2009). The First-order Markov chain is used based on the assumption that the state of a system at $t+1$ (S_{t+1}) is solely a function of the state at t (Ross, 2010). Although time-independent transition probabilities between states (i.e., $\mathrm{P}(S_{t+1} = j \mid S_t = i)$) are usually used (Thompson *et al.*, 1998; Ross, 2010), and time-dependent transition probabilities as a function of elements' age have also been proposed (Sinha *et al.*, 2009) to model the deterioration of elements. Deterioration rates of bridge elements in this case study are based on the proposed rates in IBMS (Sinha *et al.*, 2009).

4-3-1-3 Probabilistic hazards and responses

Hazards and the hazard responses of assets are the second categories of uncertainties in this study. Although hazards are rare incidents, they usually lead to enormous subsequent losses. Hazard occurrence and sampling could be modeled with the Poisson process (Li *et al.*, 2020) as shown in Eq. (3.1). The response of a bridge to an earthquake occurrence could be evaluated by the fragility curves proposed in the HAZUS (FEMA-NIBS: Federal Emergency

Management Agency (FEMA) by the National Institute of Building Sciences, 2003), as shown in Eq. (3.2). However, the parameters of the fragility curves proposed in the HAZUS govern intact assets and not deteriorated ones. In other words, this approach provides a similar probability of exceedance from damage states for both intact and degraded assets. Other studies (Ghosh and Padgett, 2009; Dong, Frangopol and Saydam, 2014) have suggested time-variant fragility curves to incorporate deterioration due to corrosion into seismic performance evaluation and finding the conditional probability of damage states in response to earthquakes. Inspired by the HAZUS methodology and without loss of generality, state-dependent fragility curves are used in this study to overcome this limitation. In this approach, the probabilities of exceedance from the deteriorated state to the collapsed state are normalized to keep the sum of probabilities equal to 1. As a result, the probability of exceeding a damage state from state S_j could be quantified as:

$$P_{S\geq S_i|IM,S_j} = \Phi\left\{\frac{1}{\beta_{S_i}}\ln\left(\frac{IM}{m_{S_i}}\right)\right\}/\Phi\left\{\frac{1}{\beta_{S_i}}\ln\left(\frac{IM}{m_{S_j}}\right)\right\},\ S_i \geq S_j \qquad (4.5)$$

The CR corresponding to a damage state can be found in Table 4.2 which maps the two systems based on their descriptions. Notably, although HAZUS – MH2.1 does not provide information regarding casualty data and losses for bridges, the methodology holds for other assets and their response that would yield different losses.

4-3-1-4 Costs volatility

Costs volatility is the third source of uncertainty in this study. The uncertainty in costs stems from factors such as fuel price and average daily traffic. The Wiener process has been extensively applied for short/long-term modeling of uncertain prices and values in finance and economics (Brennan and Schwartz, 1976; Pindyck, 1993; Ross, 2010; Hirsa and Neftci, 2013; Kim *et al.*, 2017; George and George, 2018; Kim and Lee, 2018; Capasso, Gianfrate and Spinelli, 2020). It has also been employed in the construction domain similarly (Ashuri *et*

al., 2012; Ilbeigi and Ashuri, Baabak Hui, 2014). The Wiener process is a category of stochastic processes for modeling continuously volatile market prices and indicators (Hirsa and Neftci, 2013). Consistent with these studies, it is assumed that user costs volatility follows the Wiener process with drift, Eq. (4.6), in this study:

$$v(t) = v_0 + \eta t + \sigma W_t \quad (4.6)$$

where W_t is the Wiener process, η is the drift ratio (the trend of costs) and σ is the standard deviation (volatility of costs), and v_0 is the initial value. The drift ratio and standard deviation of Eq. (4.6) can be fine-tuned and calibrated with historical data.

4-3-1-5 MRR plans and recovery actions

Maintenance, rehabilitation, reconstruction, and do nothing are four typical actions that are planned for assets in a time horizon (Hawk and Small, 1998; Thompson *et al.*, 1998; Sinha *et al.*, 2009). Figure 4.3 also shows a possible MRR plan of a bridge in this case study consisting of these possible actions (i.e., 0: do nothing, 1: maintenance, 2: rehabilitation, 3: reconstruction), represented as a 2-D vector. Recovery actions refer to a set of actions that should be undertaken after the occurrence of a hazard to restore the asset to an acceptable service level. The effectiveness of MRR activities and recovery actions were inspired by previous BMSs, such as BRIDGIT (Hawk and Small, 1998), or rationally assumed (i.e., the CR of the asset after recovery actions will be similar to that of NBI rating 8).

		0-2	**2-4**	**4-6**	**6-8**	**8-10**	**10-11**	**12-14**	**14-16**	**16-18**	**18-20**	
Element 1	[[	0	0	3	0	0	0	0	0	0	0	]
Element 2	[	0	0	1	0	0	1	0	3	0	1	]
Element 3	[	0	0	0	0	0	1	0	3	0	0	]]

Figure 4.3. MRR plan of a bridge

4-3-1-6 User costs, agency costs, and utilities

User costs MRR actions for bridges are mainly incurred because of delays in the transportation times of users and commuters. These costs can be modeled as a function of fuel price and workers' hourly wages. The user costs functions that are implemented in this study are based on the estimates provided by the Texas Department of Transportation (2020):

$$C_U = C_d + C_f \quad (4.7)$$

$$C_d = T_p \times ADT \times \left[\left(\frac{L_b}{V_a} - \frac{L_b}{V_b}\right)(1 - p_d) + \left(\frac{L_d}{V_d} - \frac{L_b}{V_b}\right)p_d\right] \times \left[p_T C_{d_T} + (1 - p_T C_{d_P})\right] \quad (4.8)$$

$$C_f = T_p \times ADT \times [L_b(1 - p_d) + L_d p_d] \times \left[p_T C_{f_T} + (1 - p_T C_{f_P})\right] \quad (4.9)$$

where C_U is the total user costs, C_d is the set of costs due to travel delay, C_f is costs due to excessive fuel consumption, T_p is project duration, ADT is average daily traffic, L_b is the length of the bridge or MRR projects, L_d is the length of detour (alternate road), V_a is average speed prior to construction, V_b is the average speed during construction, V_d is the average speed in the detour, p_T is the truck percentage, p_d is the percentage of drivers that would use detours, C_{d_T} and C_{d_P} are values of travel time for trucks and personal vehicles, C_{f_T} and C_{f_P} are marginal costs of trucks and personal vehicles fuel burn. Further details regarding costs and other parts of user costs formulas can be found in (TexasDOT, 2020).

Agency costs refer to the direct monetary resources that must be invested in the maintenance, rehabilitation, or reconstruction of bridges (or assets in general). These agency costs could be formulated as a function of the design type of the bridges, element type, material, and area or volume of the project. Sinha et al. (2009) proposed using the Cobb-Douglas production function (Nicholson and Christopher, 2011) for estimating the agency costs:

$$C_A = A_r \times L_b^{\alpha_r} \times W_b^{\beta_r} \quad (4.10)$$

where c is estimated project costs, A_r, α_r, β_r are regression coefficients, L_b and W_b are the length and width of bridges. Given the type of project, elements type, and materials, regression coefficients in Eq. (4.10) could differ from one another. These regression coefficients and further details could be found in (Sinha *et al.*, 2009).

Utility theory has been widely used to measure how appealing an MRR plan is to the agencies and decision-makers. In this study, the utility of MRR actions regarding the deck, substructure, and superstructure of bridges are (Bai *et al.*, 2013):

$$u_{DC} = 122.75 \times (1 - e^{-0.19x}) \quad (4.11)$$

$$u_{SP} = 119.13 \times (1 - e^{-0.203x}) \quad (4.12)$$

$$u_{SB} = 119.49 \times (1 - e^{-0.202x}) \quad (4.13)$$

where u_{DC}, u_{SP}, u_{SB} are the utility of deck, superstructure, and substructure with a CR of x. Consequently, the utility of actions can be quantified as:

$$U = u_2 - u_1 \quad (4.14)$$

where u_2 and u_1 are the utility of the element after and before conducting an MRR action. Multi-attribute utility theory is usually used to combine several utilities into one to simplify the optimization process (Bai *et al.*, 2013; Frangopol, Dong and Sabatino, 2017). Accordingly, the weighted sum of bridge elements' utilities with equal weights is used as the total utility in this case study.

4-3-1-7 The LCCA module

MCS is usually used to consider the uncertainties and calculate the expected values of outcomes (i.e., user costs, agency costs, and utilities). Although other factors such as reliability, sustainability, and risk could also be quantified and analyzed for each MRR plan in a life cycle (Frangopol, Dong and Sabatino, 2017), this study focuses on the average of the user costs, agency costs, and utilities without loss of generality. The current implemented LCCA module in GIAMS can yield agency costs, user costs, and the utility of implementing

a proposed MRR plan in the investment horizon of a bridge/network. Details of the computational steps in the LCCA module and relations among the implemented models in the case study are provided in Algorithm E1 in Appendix E.

4-3-2 LCCA parameters of bridges

LCCA parameters can be divided into three main groups: 1) constants which are the underlying assumptions in this case study, 2) variables which are bridge-specific parameters, and 3) MRR plans which are possible timings for conducting maintenance, rehabilitation, or reconstruction of a bridge. These parameters, including their value or range of values, are summarized in Table 4.3.

Table 4.3. Constants and variables of synthesized bridges

Parameters	**(Range of) Values**	**Role**
Number of elements	3	
Element 1	Deck	Number of elements and elements that are used in the case study
Element 2	Superstructure	
Element 3	Substructure	
Period of inspection/planning (years)	2	For developing MRR plans[1]
Discount rate (%)	3	Assumption of the case study
Horizon (years)	20	Management horizon for life cycle analysis[2]
Length (m)	(5, 1800)	Used in the computation of agency costs[3] and user costs[4]
Width (m)	(3, 60)	
Design	{1,2,3,4,5,6,7,10,11,12,14,16,21,22}	Used in computation of agency costs[3]
Vertical clearance (m)	(4, 7)	
Material	{1,2,3,4,5,6}	Used in the computation of agency costs[3] and deterioration[3]
Road class	Local, Major, Minor,	

	NHS	
ADT (vehicles/day)	(100, 400000)	
Truck percentage (%)	(0, 0.5)	
Detour length (km)	(1,100)	
Maintenance duration (days)	(10, 60)	
Rehabilitation duration (days)	(120, 240)	
Reconstruction duration (days)	(300, 540)	
Traveling speed before project (km/hr)	(40, 90)	Used in computation of user costs[4]
Traveling speed during the project (km/hr)	(15, 35)	
Drift ratio in the user costs model[3]	(0.01, 0.1)	
Standard deviation in the user costs model	(0.01, 0.1)	
Detour usage percentage (%)	(0, 0.99)	
HAZUS class	HWB {1, 3, 5, 8, 10, 12, 15, 17, 22}	Used in finding fragility curves parameters and the response of assets[5]
Soil type class	A, B, C	
Skew angle (degree)	(0, 45)	
Number of spans	(1, 60)	
Earthquake occurrence rate (events/year)	(0.001, 0.1)	
Earthquake magnitude distribution: Lognormal mean	(3, 5)	Used for generating hazards[6]
Earthquake magnitude distribution: Lognormal standard deviation	(0.01, 2)	
Deck condition	{9,8,7,6,5,4}	Condition of elements upon which deteriorated

Deck age (years)	(1, 90)	conditions[3] and response of assets to hazards[5] is found, recovery actions and MRR plans are conducted
Deck material	{1, 2, 3, 8}	
Superstructure condition	{9,8,7,6,5,4}	
Superstructure age (years)	(1, 90)	
Substructure condition	{9,8,7,6,5,4}	
Substructure age (years)	(1, 90)	

Note: 1- (FHWA (Federal Highway Administration), 2012), 2- (Hawk and Small, 1998), 3, (Sinha *et al.*, 2009), 4- (TexasDOT, 2020), 5- (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003), 6-(McGuire, 2004)

4-3-3 Sampling bridge LCCA parameters and results

Bridges' characteristics, MRR plans, and environmental factors are randomly synthesized to generate sample bridges based on the Indiana bridge network available in NBI. After conducting LCCA for each sample bridge, the life cycle analysis results, as well as other related parameters, are stored in a dataset for training ML models. Each synthesized bridge with its random MRR plan is a point in the feature space for the ML model. Accordingly, more than 1.4 million synthesized bridges (samples) with random MRR plans were sampled and analyzed. Considering deterioration, earthquakes, and user costs as the main sources of uncertainties, approximately 1000 simulations were required to reach a 95% confidence interval for the LCCA results. This estimation was derived based on the central limit theorem which states the average of simulations results follows the normal distribution with an average of $\mu_{\bar{x}}$ and standard deviation of $\frac{\sigma}{\sqrt{n}}$ for n iterations. Accordingly, the confident interval, Eq. (4.15), and a minimum number of iterations, Eq. (4.16), could be derived by (Law and Kelton, 2000):

$$\pm \mathrm{Z}\frac{\sigma_0}{\sqrt{n}} \quad (4.15)$$

$$n \geq \left(\frac{\mathrm{Z} \times \sigma_0}{\mathrm{E}}\right) \quad (4.16)$$

where σ_0 are an initial estimate of standard deviation and E maximum allowable error. In this study, 1% of the initial estimate of the mean was set as the maximum allowable error (E). Table 4.4 provides a statistical summary of the synthesized LCCA target value results (i.e., covering user costs, agency costs, and utility).

Table 4.4. Statistical summary of user costs, agency costs, and utility

	Mean	**Min**	**25%**	**50%**	**75%**	**Max**
User costs ($1000)	111693.8	10.0	615.0	4,681.4	62,383.1	1,995,216.7
Agency costs ($1000)	9481.1	60.0	2,352.1	5,940.1	12,534.1	59,968.2
Utility	31.6	3.0	21.8	30.8	40.8	70.0

4-3-4 Estimating LCCA of bridges with DNN

4-3-4-1 Data preprocessing

The LCCA parameters must be normalized, encoded, and pruned to be able to be fed to ML models because of redundant parameters, nominal parameters, ordinal parameters, and differences in the ranges of continuous variables. First and foremost, redundant variables (constants for all samples) should be eliminated to reduce dimensionality while maintaining useful information from datasets. Constant parameters such as the number of elements are removed since they are shared among all samples and will have an adverse effect on the ML model training. More importantly, not all variable parameters equally affect the three main outputs (i.e., user costs, agency costs, and utility) of the LCCA. For example, detour length affects the user cost while it does not affect agency costs and utilities. To reduce dimensionality and consequently prevent the learning models from overfitting, three different subsets of the dataset were created for user costs, agency costs, and utilities with redundant features removed. These three datasets contain several parameters such as CRs in common and some parameters exclusively. Table 4.5 summarizes the parameters that are excluded from each dataset.

Table 4.5. Excluded parameters from each dataset and learning models

User costs model	Agency costs model	Utility model
Width	ADT	Length
Design	Truck percentage	Width
Vertical clearance	Detour length	Design
	Maintenance duration	Vertical clearance
	Rehabilitation duration	ADT
	Reconstruction duration	Truck percentage
	Traveling speed before the project	Detour length
	Traveling speed during the project	Maintenance duration
	The drift of the user costs	Rehabilitation duration
	The volatility of the user costs	Reconstruction duration
	Detour usage percentage	Traveling speed before the project
		Traveling speed during the project
		The drift of the user costs
		The volatility of user costs
		Detour usage percentage

Second, one-hot encoding is used to convert nominal parameters (e.g., material, road type, HAZUS classification) into a string of binaries. Each nominal parameter with k categories is converted to $k-1$ binary parameters by one-hot encoding. Also, MRR actions for each year were converted from categorical to binary variables in a different manner. Do nothing, maintenance, rehabilitation, and reconstruction were first converted to integers, 0, 1, 2, 3, respectively. Then these integer values were converted to binary values (e.g., 3 was converted to 1, 1). As a result of encoding MRR actions and one-hot encoding of other categorical variables, 32 parameters of simulation and bridges' characteristics and 30 parameters of a 20-year horizon MRR plan for three elements were converted to a total of 122 normalized and

binary parameters. Finally, since the range of continuous variables varies, they should be normalized to a range between 0 and 1 for ML training:

$$N(\mathbf{X_i}) = \frac{\mathbf{X_i} - \min(\mathbf{X_i})}{\max(\mathbf{X_i}) - \min(\mathbf{X_i})} \quad (4.17)$$

where $N(\mathbf{X_i}), \min(\mathbf{X_i}), \max(\mathbf{X_i})$ are the normalization, minimum, and maximum of the parameters $\mathbf{X_i}$. Similarly, normalization should be applied to ordinal parameters (e.g., CRs).

4-3-4-2 Hyperparameter tuning

Following the work of (Asghari, Leung and Hsu, 2020), DNN hyperparameters including, but not limited to, optimization algorithm, activation functions, cost function, type, and the number of layers were determined in this study and are summarized in Table 4.6.

Table 4.6. Hyperparameters of the DNN models

Hyperparameters	**Status**
Number of hidden layers	*3*
Number of nodes in each layer	*500*
Input layer activation function	*tanh*
Hidden layers activation function	*ReLu*
Output layer activation function	*linear*
Optimizer	Adam
Cost function	MAPE
Epoch	1000
Batch size	4096
Slicing proportions (Training, cross-validation, and test set)	90:5:5
Weight regularization parameter and type	0.000001, L2

A number of these hyperparameters are set given the nature of the problem. For example, the linear function is suggested as the final layer activation function for regression tasks (Ng, 2018). Some of the hyperparameters, including Adam optimizer as the optimization function (Kingma and Ba, 2015), *ReLu* as the hidden layer activation functions (Xu *et al.*, 2015), *tanh*

as the input layer activation function (Ng, 2018), are reportedly recommended in the literature based on their superior performance in comparison to their counterparts. Considering the convergence speed and prediction accuracy, batch sizes are suggested to be relatively small and be a power of 2 (e.g., $2^1, 2^2, \dots, 2^j$) (Keskar *et al.*, 2017). The slicing proportions are arbitrary values that are set based on dataset sizes. To further illustrate, smaller test size portions can be used for big datasets (Ng, 2018). Early stopping as a regularization technic can be used to terminate optimization when there is no improvement in the accuracy of prediction results on the cross-validation set (Ng, 2018). To this end, a large number of epochs (iterations of optimization) is used not to terminate the optimization before the early stopping technic does. Starting from smaller neural networks with few hidden nodes and one hidden layer, different structures should be trained and tested to minimize and improve prediction accuracy (reduce the variance problem). After finding a structure for the neural networks that can yield acceptable prediction results (with low variance problem), L1 or L2 regularization technics can be used to mitigate possible overfitting problem (minimize difference between the prediction results on test set and train set) (Ng, 2018). Depending on the type of training goals, cost function considerably impacts prediction accuracy and training time. In this study, for example, the range of user cost values is relatively large. If MSE or MAE are chosen as the cost function of DNN to model user costs, the model would try to fit on larger values to attain the lowest possible MSE at the cost of neglecting smaller values. Therefore, by normalizing errors, MAPE would be a better choice for the cost function of the DNN model in this study. To compare the effectiveness of cost functions in this study, Figure 4.4 depicts the MAPE of predictions for 3 different cost functions after 100 epochs of training.

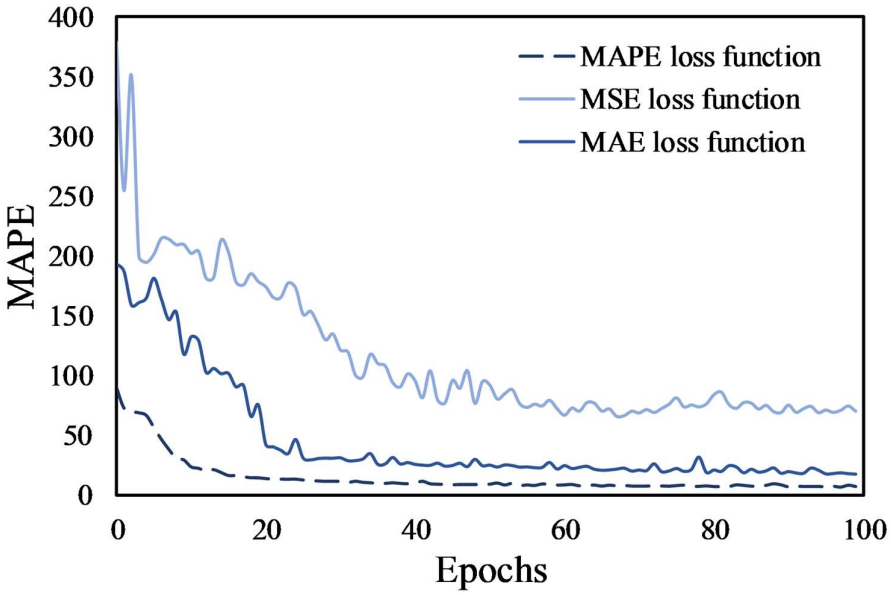


Figure 4.4. MAPE of three cost functions

4-4 Results and Discussion

Calculation of each LCCA using the GIAMS platform and considering the three sources of uncertainties takes 5.3 seconds. The LCCA of the synthesized bridges were analyzed and evaluated using an Intel(R) Xeon(R) E5-2697 CPU, 128 GB RAM, with 72 logical processors. Through leveraging parallel processing and using all 72 processors, the whole bridge sampling session took nearly 31.5 hours. Then the dataset was normalized, encoded, and pruned to form three datasets for training three models for user costs, agency costs, and utility. The neural networks training process was conducted by GPU NVIDIA Quadro P620 and with specialized libraries required for GPU training including CUDA 10.1, and cuDNN 7.4. The training session took approximately 57 minutes.

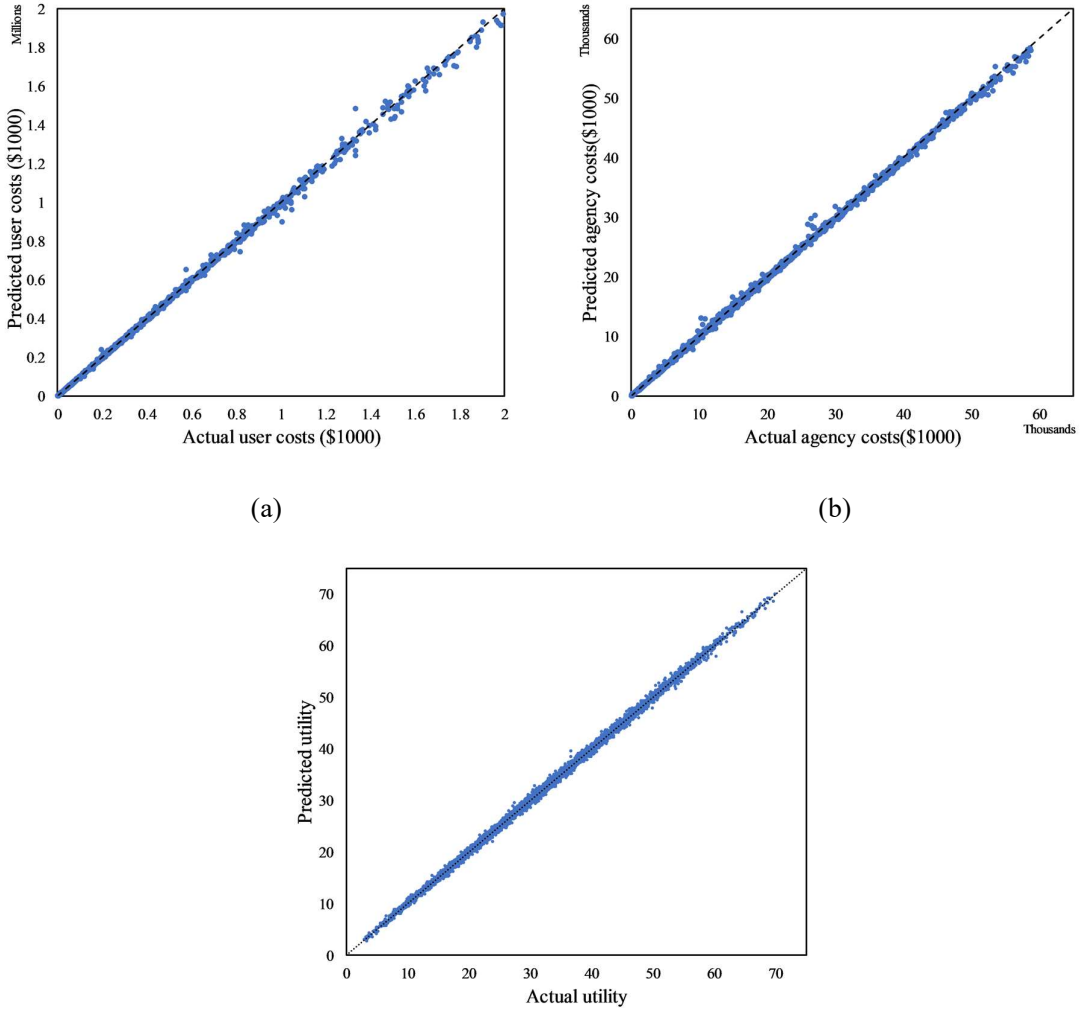
Other ML models (i.e., DTs, RF, shallow neural network, and linear regression) are trained to compare their results with that of the trained DNN model. Following previous studies (R. Wang *et al.*, 2020), the hyperparameters of these models were set as follows: A) DT: maximum branching depth = 5, minimum samples in each leaf for splitting = 2,

minimum samples to be in each leaf = 1, B) RF: number of trees = 500, number of features to look for when splitting = all features, maximum branching depth = 5, minimum samples in each leaf for splitting = 2, minimum samples to be in each leaf = 1, C) Shallow neural network: similar to the proposed DNN but with only one layer, D) Linear regression: ordinary least squares (OLS) regression. Notably, the high computation time of the support vector machine and k-nearest neighbors on large datasets made them infeasible for evaluation in this study. Since the ranges of the user and agency costs are large, the MSE values are misleading and vague for assessing the prediction performance of the regression models in this study. Therefore, R-squared and MAPE among other prediction accuracy metrics are provided. The results of the regression analyses for all the models trained on the test sets are summarized in Table 4.7, and the corresponding graphs for visual validation of the regression analysis results are also provided in Figure 4.5.

Table 4.7. Regression analysis report on the test set

Model	**ML model**	**R2**	**MAPE**
User costs model	Linear regression	0.62	20185%
	DT	0.79	58.30%
	RF	0.81	57.99%
	Shallow neural network	0.23	42.66%
	DNN	0.99	1.55%
Agency costs model	Linear regression	0.73	179.31%
	DT	0.77	37.68%
	RF	0.80	35.62%
	Shallow neural network	0.75	13.82%
	DNN	0.99	1.27%
Utility model	Linear regression	0.55	32.77%
	DT	0.77	17.91%

RF	0.82	16.75%
Shallow neural network	0.99	2.99%
DNN	0.99	1.08%



(a) (b)

(c)

Figure 4.5. Actual vs. predicted values on test sets of (a) user costs, (b) agency costs, and (c) utility

With an R2 of more than 0.98 and MAPE of less than 2% in all models, the numerical results of the regression analysis are satisfactory. These results could also be visually validated in Figure 4.5, where the predicted values are drawn against actual values for user

cost, agency cost, and utility. In addition, based on the results shown in Table 4.7, DNN outperformed other models by a large margin, demonstrating it to be a viable approach.

Despite the applicability and acceptable prediction results, DNN models have resulted in a small number (less than 1%) of predictions with large prediction errors. Further analysis of the assets corresponding to large prediction errors revealed that most of them had short lengths with an average length of 180 meters. Another set of these synthesized assets with large errors was associated with unrealistic width to length ratios. One of the underlying reasons behind this issue would have been the inaccuracy of MCS on unnormal assets. All in all, the applicability of the DNN models on realistic bridges was generally acceptable with a MAPE of less than 5%. If further conditions are applied for generating synthesized assets and more samples were generated in the MCS, the prediction results would theoretically not suffer from this issue. Apart from the need for training more accurate models and given the black-box nature of most ML models, practitioners should conduct a series of risk analyses on the outcome of the ML models before taking decisions. Although this characteristic of ML models makes them less trustworthy, they can still be considered as accelerated methods for estimation of LCCA results by conducting risk analysis and taking further measures when DNN models would lead to larger errors.

The computation time of the LCCA module in GIAMS with three sources of uncertainties, as well as that of the estimators, is provided in Table 4.8.

Table 4.8. The computation time of regular LCCA and estimators

Method	Time (Seconds)
LCCA sampling	~5.3
LCCA estimation for one asset	~0.0004
LCCA estimation for 200 assets (average for each one)	~0.00006

Although the data synthetization and training of the models are relatively time-consuming during the training phase, the LCCA estimation is far less time-consuming during the analysis phase. After completing the overhead computation time for sampling and training, the trained DNN model can estimate the LCCA results and yield similar outcomes with an acceptable range of errors 5 order of magnitudes faster than the regular MCS method. This trade-off is especially beneficial in terms of computational time if LCCA is to be conducted millions of times in the optimization procedure. Drawing upon the previous discussion on the Texas highway bridges, computation for finding the optimal MRR of each bridge could theoretically be reduced to approximately 105.5 hours from 6 months (assuming optimization by GA with 200 generations, 200 individuals in each generation, 50,000 assets, and 12 computational processors).

Sensitivity analysis is also conducted to analyze and compare the effect of the input parameters on the final results. Although it can also be conducted on the optimization modules, the accuracy of the prediction models is high enough to make them reliable for this purpose. In fact, there is a need for further study and analysis and perhaps other modules for a better understanding of the input parameters on the final results. Although this lies beyond the scope of this study, a brief analysis based on the feature importance analysis based on random permutation proposed by (Breiman, 2001) is conducted. The result of this analysis is summarized in Table 4.9. This table shows the top 10 important features that affect the results of LCCA analysis.

Table 4.9. Summary of the sensitivity analysis on the outcomes of prediction models – Top 10 parameters

User costs	**Agency costs**	**Utility**
detour_usage_percentage	length	substructure_cond

ADT	width	superstructure_cond
speed_after	actions on elements	deck_cond
drift	material	actions on elements
recon_duration	hazus class	occurrence_rate
rehab_duration	occurrence_rate	hazus_class
truck_percentage	design	road class
length	skew_angle	deck_age
maint_duration	deck_cond	superstructure_age
actions on elements	n_spans	substructure_age

The effect of these parameters on the LCCA results is completely in line with the theory behind the formulas and models representing user costs, agency costs, and utilities of actions. That said, further tools and models are required to be implemented in the GIAMS to inform decision-makers about the effect of these models on each asset separately.

The proposed methodology, i.e., estimation of LCCA results using ML models, could be used across different domains of asset management to reduce the computational time of life cycle optimization. Complex models, different sources of uncertainties, and their consequently large computation time for LCCA are the main barriers to upscaling advanced LCCA frameworks for application to large networks. This methodology could tackle this limitation and be applied to various types of assets such as pavement, railways, and buildings. Notably, the number and range of features affect the asset synthesizing (data augmentation) step. If the management horizon increases from 20 years to 40 years and 4 elements are considered instead of three, 100 new features will be added to the dataset. The issue of dimensionality affects both the LCCA and life cycle optimization, making them even further unfeasible for use in NL-IAM. However, the proposed methodology would only require more synthesized samples so that the ML model could accurately be trained.

Chapter 5 Upscaling complex project-level infrastructure intervention planning to network assets

This chapter advances a novel methodology in the infrastructure management literature to estimate the outcome of MRR optimization algorithms (i.e., MRR plans) by a trained ML algorithm instead of directly conducting optimization. The ML model is trained on an augmented dataset containing multiple datapoints with an input vector (i.e., asset characteristics, MCS parameters) and an output vector (i.e., optimal MRR plans). The optimal MRR plans are derived from maximizing/minimizing a user-defined function based on expected LCCA results using a heuristic optimization algorithm. The expected LCCA results are computed using MCS taking into account several sources of uncertainties such as hazards, market conditions, and deterioration.

ML and AI algorithms are capable of learning complex systems, phenomena, and activities such as computer vision and natural language processing. Various types of ML algorithms have been widely used for developing decision-making tools and frameworks both in literature and real life (e.g., Ashkan et al. 2020; Evan et al. 2020; Jin and Jiansong 2019; Pouya et al. 2020; S. et al. 2021). However, the application of ML techniques has been limited to learning complex models in asset management instead of estimating the outcome of the whole decision-making system. GIAMS, an open-source asset management infrastructure management system (at: https://github.com/vd1371/GIAMS (Asghari and Hsu, 2020)), was used as the computation tool for conducting the asset management analyses and sampling in this study. The codes for sampling and ML training are also available online (at: https://github.com/vd1371/XProject, (Asghari, 2020) and https://github.com/vd1371/EstOpt (Asghari, 2021)). Given the importance of bridges in infrastructure systems and the percentage of structurally deficient ones (e.g., American Society of Civil Engineers 2021),

this chapter uses the BMS and the Indiana network of bridges as a case study to highlight the extent of capabilities of the proposed methodology. The results of this chapter demonstrate considerable improvement in the computation time required for accurately estimating MRR optimization results without compromising complex models. This chapter contributes to the body of knowledge by proposing a novel methodology for reducing the computation time of MRR optimization and an explicit set of procedures for training an ML algorithm on a semi-synthesized dataset. Using the proposed methodology, practitioners and decision-makers can derive estimates of optimal MRR plans considering various sources of uncertainties and complex models (e.g., probabilistic hazards and assets responses, deterioration, the uncertain costs) in far less time. The proposed methodology could be similarly applied to other types of asset management systems for similar purposes. Consequently, asset managers can reach more accurate results by incorporating models to fully address the complexities in uncertain phenomena for all assets in a network within a feasible amount of time.

This chapter is structured in the following sections. First, the key concepts and main steps of the methodology section are discussed, followed by a case study based on more than 4,600 bridges in Indiana, US. Next, the results of the case study and discussion about the computation time and accuracy of the developed model are discussed.

5-1 Methodology

Figure 5.1 depicts the steps of the proposed methodology in an abstract form. These steps comprise a distinct set of actions that should be conducted for training an ML algorithm to predict the outcome of MRR optimization frameworks. The provided methodology is a general and flexible guideline for estimating optimization results. This theoretically well-founded and flexible methodology is consistent with the existing literature. The flexibility of the proposed methodology allows its extension to other types of intervention actions

optimization in infrastructure management and maintenance planning (i.e., None of the proposed methodology's steps, as shown in Figure 5.1, is problem-specific)

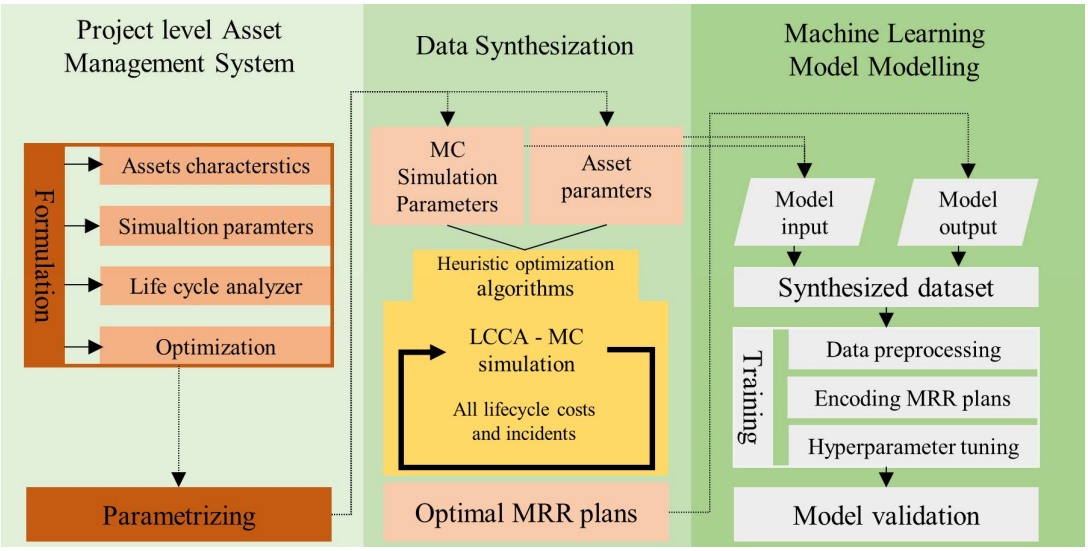


Figure 5.1. The abstract flowchart of the proposed methodology

The core idea behind this methodology is training an ML algorithm on the solutions of thousands of different optimization problems (i.e., near-optimal intervention timings in the life cycle) based on semi-synthesized assets. In this manner, an ML model is trained on a dataset containing thousands of data points (the input and output vectors). The input vector may contain variables such as asset characteristics and MCS parameters, and the output vector represents the optimal MRR plan derived by an IAM system. This methodology is briefly explained in Algorithm **5.1**.

Algorithm 5.1 Pseudo-code for sampling optimization results (data points)

1: **Input:** Asset's characteristics and their ranges

2: **Input:** MCS parameters and their ranges

3: **D:** empty dataset // A holder for datapoints

4: **While** not enough_samples: // A loop for creating new samples

5: $\mathbf{A_p}$**:** generate a new set of asset characteristics

6: $\mathbf{S_{MCS}}$: generate a new set of MCS parameters

7: $\mathbf{M}^*$: find optimal MRR plan given $(\mathbf{A_p}, \mathbf{S_{MCS}})$ // using the PL-IAM system with a heuristic optimization algorithm and MCS: **Algorithm** 5.2

8: Add $(\mathbf{A_p}, \mathbf{S_{MCS}}, \mathbf{M}^*)$ to **D** // add to the dataset

9: **return D**

10: $\widehat{\mathbf{M}} = \mathrm{f}(\mathbf{A_p}, \mathbf{S_{MCS}}) + \epsilon_r$ //Train an ML model (f) using **D** – Estimate **M** using $(\mathbf{A_p}, \mathbf{S_{MCS}})$ with f as the estimator and ϵ_r as the prediction error

11: **Validate the trained model**

Ideally, the trained ML algorithm can yield optimization results (i.e., the output vector), based on the asset characteristics and MCS parameters (i.e., the input vector), perfectly similar to that of the optimization algorithms. It assumes that aleatory uncertainties (i.e., irreducible uncertainties because of the inherent uncertain characteristics of problems) do not usually exist in optimization problems formulation. In practice, however, heuristic algorithms might yield near-optimal solutions, which would alter the performance of ML algorithms. Nevertheless, the trained ML algorithms are theoretically capable of learning complex phenomena given sufficient observations and predicting the behaviors and results of the phenomena in a short amount of time. The proposed methodology in this study provides the opportunity of upscaling complex computations from project-level to network-level, which has been impractical heretofore (Frangopol, 2011).

5-1-1 Project-level infrastructure asset management (PL-IAM)

PL-IAM refers to the process of optimizing the timing and extent of prospective interventions (e.g., maintenance actions) in the life cycle of an asset with respect to limited resources and performance limitations. Maximizing the utility of intervention actions given a limited budget is an example of such an optimization process:

$$maximize \quad E\left[\sum_{t=0}^{T}\sum_{p\in P_i} U_{pt}X_{pt}\right] \tag{5.1}$$

With respect to:

$$\sum_{p\in P_i} C_{A_{pt}}X_{pt} \le B_{d_t}$$

$$X_{pt} \in \{0,1\}$$

$$\sum_{p\in P_i} X_{pt} = 1, \qquad for\ each\ year\ t$$

where X_{pt} is a binary value representing presence/absence, U_{pt} is the utility, $C_{A_{pt}}$ associated with project p (e.g., do nothing, maintenance, rehabilitation, reconstruction) for year t, and B_{d_t} is the budget limitation. Although this methodology is described through maximizing utility of actions, minimizing associated risks, maximizing reliability, or minimizing user costs are other types of objectives that could be used without loss of generality.

Depending on the nature of the problem, the complexity of the models, and the level of uncertainties, mathematical optimization or heuristic optimization methods could be used to solve for optimal or near-optimal solutions. When assets are subject to various sources of uncertainties, MCS is usually used to evaluate the expected costs and utilities of intervention actions. Finally, heuristic algorithms such as the GA can be used to find near-optimal solutions based on the LCCA results calculated by MCS or estimated by a trained ML algorithm. The pseudo-code of this methodology is explained in **Algorithm** 5.2**.**

Algorithm 5.2. Pseudo-code of project-level optimization with GA and Monte Carlo simulation

---

1: **Input:** Asset's characteristics ($\mathbf{A_p}$)

2: **Input:** MCS parameters ($\mathbf{S_{MCS}}$)

3: **while NOT** termination conditions**:** // Optimization continues until certain conditions

are met

4: **if** first_genration:

5: Randomly create first-generation → **G** // **G**: a randomly generated set of possible MRR plans

6: **else:**

7: **G = create_new_generation(G)** // **G**: a new set of MRR plans generated based on the previous generation

8: **for each** $\mathbf{M}_i \in \mathbf{G}$: // iterate over all MRR plans in the generation

9: **for** $j \in \{1, 2, \dots, N_s\}$ **do**: // N_s: Number of MCS samples

10: **SC** = a sample scenario // Generate a random sample of all incidents in a lifecycle

11: $\mathbf{R}_{\mathbf{LCCA_{ij}}} = \text{LCCA}(\mathbf{A}, \mathbf{S_{MCS}}, \mathbf{M_i}, \mathbf{SC})$ // Find the objective value function as a result of LCCA and store it in $\mathbf{R}_{\mathbf{LCCA_{ij}}}$

12: **end for**

13: **find** $\bar{\mathbf{R}}_{LCCA_i}$ // Find the average of LCC

14: **end for**

15: $\mathbf{M}^* = argmax(\bar{\mathbf{R}}_{LCCA_i})$ // the best MMR plan with the highest objective value

16: **end while**

17: **return** $\mathbf{M}^*$

Although MCS has been widely used for the calculation of LCC in this process, Asghari, *et al.* (2021) proposed using trained ML algorithms (e.g., DNN) for estimation of LCCA results with MCS (lines 9 to 12 of Algorithm 5.2) to reduce computation time.

The above-described methodology has the flexibility to incorporate a variety of complex models (e.g., probabilistic hazards and assets' responses, non-deterministic deterioration models). In addition, it can provide near-optimal solutions, is easily understood, and can be implemented by practitioners and researchers. Given the extent and complexity of different

problems, the computation time of this process can range from a few minutes to several hours (Kim and Frangopol, 2018).

5-1-2 Parametrizing PL-IAM

All existing parameters in a PL-IAM problem must be meticulously categorized before generating data points and training the ML model. To put this concept simply, all existing parameters in the optimization parameters should be sorted and analyzed for MRR optimization and estimation by a trained ML model. Four main categories of parameters usually exist in a PL-IAM: 1) asset characteristics ($\mathbf{A_p}$): length, width, material, age, response models, elements, etc., 2) MCS parameters ($\mathbf{S_{MCS}}$): hazard characteristics, market volatility, inflation rates, management horizon, etc., 3) LCC and utilities ($\mathbf{R_{LCCA}}$): all expected costs and utilities as a result of MRR actions, 4) MRR plans ($\mathbf{M}$): different types of maintenance actions. All these parameters should be vectorized before sampling data points and training ML models. In a project-level optimization problem, asset characteristics ($\mathbf{A_p}$) and MCS parameters ($\mathbf{S_{MCS}}$) are input vectors upon which LCC utilizes ($\mathbf{R_{LCCA}}$) are evaluated for finding optimal MRR plans ($\mathbf{M}^*$). This concept is also briefly depicted in Figure 5.2.

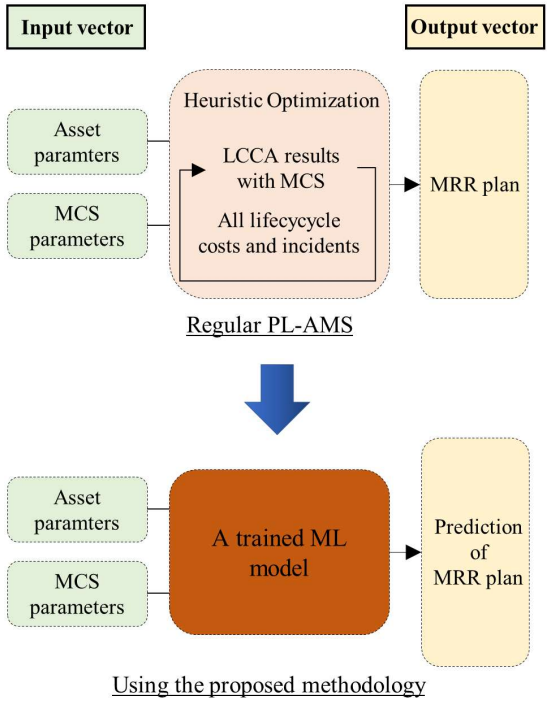


Figure 5.2.Diagram of estimating optimization solutions with trained ML models

5-1-3 Sampling datapoints

ML models are data-driven algorithms, i.e., thousands of data points are required for training and developing accurate ML prediction models. Generally speaking, the bigger the size of the datasets, the more accurate prediction results could be achieved (Ng, 2018). LCCA results ($\mathbf{R_{LCCA}}$) and optimal MRR plans ($\mathbf{M}^*$) are derived based on theoretically well-established models and optimization algorithms. However, since limited number of assets are usually available, all or a subset of asset characteristics ($\mathbf{A_p}$) and MCS parameters ($\mathbf{S_{MCS}}$) can be synthesized based on available assets. Some of these parameters could be considered constant (e.g., inflation rate, management horizon) while other parameters (e.g., material, age) could be generated, based on previous studies or managerial requirements, for developing semi-synthesized assets. Then, the optimal MRR plan of each synthesized asset could be obtained by solving an optimization problem given the results of LCCA. Although the abstract

flowchart of the datapoints sampling process is demonstrated in Figure 5.1 and Algorithm **5.1**.

When data collection for expanding datasets is expensive or impossible in computer science problems, such as MRI images, data augmentation techniques are employed for expanding datasets based on available data points (Redmon *et al.*, 2016). Data synthetization for estimation of optimal MRR actions can resemble data augmentation techniques. The number of required data points during the data augmentation session is heavily reliant on the complexity of optimization problems, selected ML models, and sampling strategy. Vanilla sampling could lead to relatively more non-realistic data points. As ML models are capable of learning complex phenomena with any degree of complexity, relatively unrealistic asset characteristics and MCS parameters would not nullify the proposed methodology. However, this would lead to more required data points, a bigger storage capacity, and a longer training time. Other techniques such as generating new data points based on a cluster of similar assets or adding Gaussian-Noise to the current data points would generate less unrealistic data points. Further details regarding these sampling techniques could be found in (Chawla *et al.*, 2002; Branco, Torgo and Ribeiro, 2019).

5-1-4 Estimating optimal plans

An ML model can be trained on the semi-synthesized datasets derived by sampling data points. Eventually, the abstract flowchart of PL-IAM depicted in Figure 5.1 can be merged into the demonstrated diagram in Figure 5.2. Estimation of optimal MRR actions has specific characteristics. First, MRR plans can be transformed/encoded to binary or decimal vectors (Asghari and Hsu, 2021). Regardless of the encoding techniques, the target space of this optimization problem is discrete. Second, MRR plans provide recommended intervention timings within a management horizon. Third, the intervention actions at the t^{th} year for all

assets could be predominately dedicated to a specific type of action (e.g., regular maintenance) and sporadically designated to other intervention actions (e.g., reconstruction). All in all, estimation of MRR optimization is an imbalanced multi-output and multi-class (IMOMC) classification problem from an ML point of view.

The results of this classification problem can be further validated by classification metrics such as accuracy, Eq. (5.2), precision, Eq. (5.3), recall, and f1-score, Eq. (5.4), on unobserved data points.

$$Accuracy = \frac{TP_j + TN_j}{TP_j + FP_j + TN_j + FN_j} \quad (5.2)$$

$$Precision = \frac{TP_j}{TP_j + FP_j} \quad (5.3)$$

$$Recall = \frac{TP_j}{TP_j + FN_j} \quad (5.4)$$

$$F1\ score = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \quad (5.5)$$

Where $T = True, F = False, P = Positive, N = Negative$. These values for this multiclass problem are depicted in Figure 5.3.

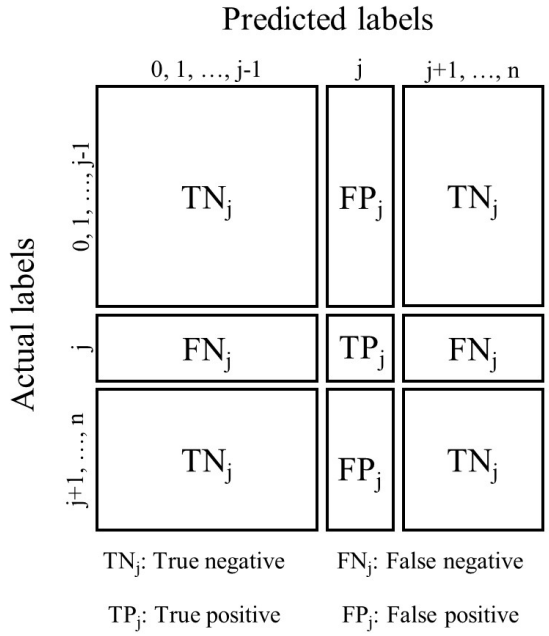


Figure 5.3. Confusion matrix of multiclass classification

5-1-4-1 Advantages and disadvantages of different ML algorithms

Training and prediction of different ML algorithms on big IMOMC classification problems are associated with upsides and downsides. First and foremost, the synthesized data is expected to contain thousands of data points. As a result, training classification algorithms with computation complexity as a function of the number of data points, such as k-nearest neighbor and support vector machine would be impractical. Second, ML algorithms that could provide satisfactory results directly on imbalanced datasets are preferred. The considerably huge required computation time for balancing imbalanced datasets with common algorithms, such as synthesize minority over-sampling technique (Chawla *et al.*, 2002), renders ML algorithms sensitive to imbalanced datasets impractical. Third, one-hot encoding or one-vs-all technique must be used to make some ML algorithms such as DNN applicable in this problem. However, one-hot encoding expands the target space and the one-vs-all technique requires a distinct model for each label. In other words, the implementation of these techniques in this problem could lead to an inefficient training session or inaccurate results. Considering the abovementioned discussions, DT-based ML models are an appropriate choice because they can be directly trained on big, imbalanced, and multilabel datasets.

5-1-4-2 DT-based ML algorithms

DT-based models, in this study, refer to ensemble methods such as RF (Breiman, 2001) that are trained and developed based on several simpler forms of DT models such as classification and regression trees (CART) (Breiman *et al.*, 2017). During the training session of CART, root nodes (features) are optimally split into leaf nodes for making predictions in a recursive manner until certain criteria are met. These criteria could be: 1) if the expanded tree as a

result of branching and splitting reaches a pre-defined maximum depth (d), 2) if remaining observations in leaf nodes are fewer than a certain number (N_l), 3) if observations in a root node are fewer than a minimum required number of data points for split (N_r). Selection of features and split points at each branching point is conducted in a way to minimize Gini impurity, Eq. (5.6), or maximize information gain by reducing entropy, Eq. (5.7).

$$G = 1 - \sum_{i=1}^{n} p_i^2 \quad (5.6)$$

$$I = -\sum_{i=1}^{n} p_i \log_2 p_i \quad (5.7)$$

Where p_i is the percentage of label i in each leaf node. Aiming to solve the overfitting problem of CART models, pruning techniques are suggested and used to make the trained models generalizable and not sensitive to small changes in datasets.

RF model is an ensemble of several trees each of which is trained on a different subset of the original dataset independently. These subsets: 1) contain a randomly selected subset of features, 2) contain a randomly selected subset of data points with replacement (bootstrapping), 3) must have similar, but not necessarily identical, distribution. Training DTs on each subset in RF is also different from that of the CART model from another point of view. Breiman (2017) suggests pruning is not necessary for RF models given the random selection of subsets. Finally, the majority of votes cast by all trained trees could be used for making final predictions in classification problems, although summation and average of the probabilities of class labels can be similarly used (Pedregosa *et al.*, 2011). The structure of an RF model is illustrated briefly in Figure 5.4.

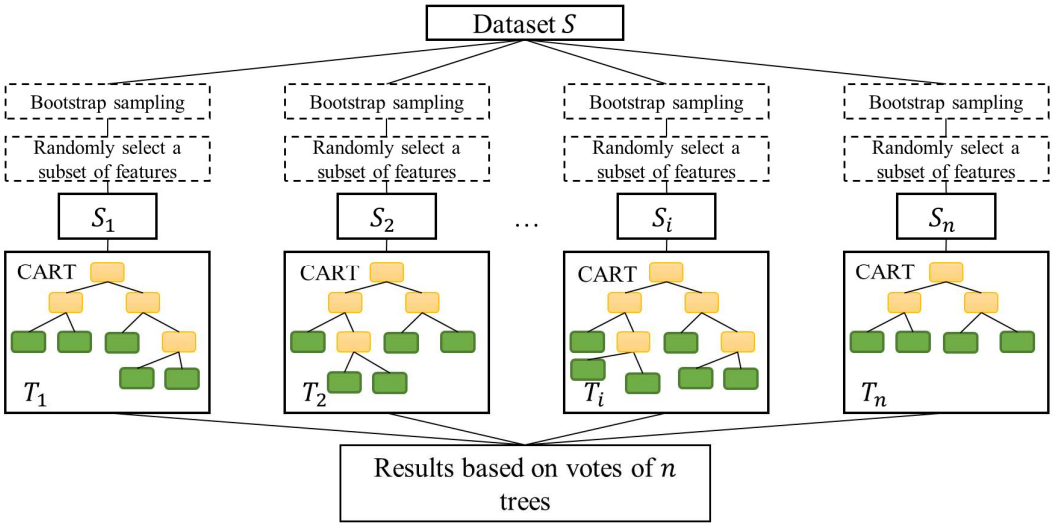


Figure 5.4. RF algorithm and structure

RF and DTs are well-known algorithms and they have been widely used in various domains of knowledge. For further details regarding these models please refer to (Breiman, 2001; Breiman *et al.*, 2017).

5-2 Case study: MRR optimization of bridges

Project-level MRR optimization of bridges is used as an example in this case study. In the following section, first, the project-level BMS that has been used in this study is briefly introduced. Then, all considered parameters for modeling phenomena governing assets are summarized. Then, the datapoint sampling procedure is discussed followed by the training procedure of RF models.

5-2-1 Project-level bridge management

The ultimate purpose of asset managers and decision-makers, in project-level bridge management, is to optimally allocate limited funds to various types of intervention actions for each bridge element in the life cycle of an asset considering various sources of uncertainties such as deterioration, hazards, and costs (FHWA (Federal Highway Administration), 2012). One of the first steps in project-level bridge management is defining the CR and finding the

corresponding deterioration rates for each element. In this study, deck, superstructure, and substructure elements are selected. Their corresponding CR is based on the national bridge inventory (NBI) of the US (FHWA (Federal Highway Administration), 2012) and the deterioration model and rates are adopted from the IBMS (Sinha *et al.*, 2009). The next step toward MRR optimization is determining hazard occurrence and bridge response modules. The earthquake has been chosen as the main source of hazard in this study. Several studies have proposed various models and intensity measures to characterize earthquake occurrences and site-specific ground shaking parameters. Following a well-known practice in earthquake engineering (McGuire, 2004; Mahsuli and Haukaas, 2013; Asghari, Kashani and Hsu, 2021), the earthquake occurrence times are modeled with the Poisson process, Eq. (3.1), and earthquake magnitudes are based on historical occurrences distribution. Bridges' responses are also found based on the guidelines and specifications in the earthquake technical manual HAZUS (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003). This concept is well discussed in Chapter 3 of this dissertation. The type of actions (i.e., maintenance, rehabilitation, and reconstruction of elements) are adopted from IBMS (Sinha *et al.*, 2009). The costs associated with user costs as a result of excessive fuel consumption and travel delay are calculated based on (TexasDOT, 2020). Encoding and formulating the MRR plans for optimization purposes is the next step in project-level management. Scenario sampling (e.g., in (Rahimi and Mahsuli, 2019)), a variant of MCS, is used to find the average of the life cycle results by instantiating one sample from all sources of uncertainties and common methodologies in LCCA (Ellingham and Fawcett, 2007). Finally, similar to the majority of studies in asset management (Chen and Bai, 2019), the GA is used for finding near-optimal MRR plans for bridges. The GA is a well-known

procedure for finding near-optimal solutions when the closed-form of an objective function is not accessible.

The objective function of the presented case study is based on the project-level example in the work of Asghari and Hsu (2021). The GA hyperparameters are set based on the guidelines provided in the literature (Grefenstette, 1986; Witt, 2013; Chicano *et al.*, 2015) with the values provided in Table 5.1.

Table 5.1. Hyperparameters of the GA

Hyperparameter	Value
Selection method	Ranking method
Crossover probability	0.8
Mutation probability	0.05
Population size	200
Number of generations	200
Number of elites	10

An example of optimal MRR plans with both decimal and binary encodings is depicted in Figure 5.5 In the decimal encoding, 0, 1, 2, and 3 denote do nothing, maintenance, rehabilitation, and reconstruction actions. These numbers are converted to their base-2 numeral form in binary encoding.

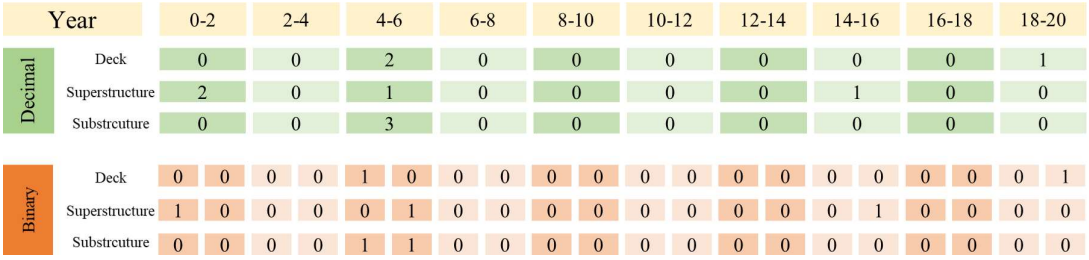


Figure 5.5. Optimal MRR examples with decimal and binary encodings

The whole project-level bridge management analysis with all computational details (e.g., life cycle analysis and the GA module) is based on the general IAM platform GIAMS. To

keep this chapter concise and to the point, details of the whole processes of project-level bridge management, as well as the corresponding references (Hawk and Small, 1998; FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003; McGuire, 2004; Sinha *et al.*, 2009; W. Ryan *et al.*, 2012; Bai *et al.*, 2013; TexasDOT, 2020), with the symbols provided in Table 5.2 are illustrated in Figure F. 1. In Appendix F. Further details are found in Chapter 3.

Table 5.2. Range and type of parameters with their corresponding pre-processing action

Parameters	Symbol	(Range of) Values	Data type	Pre-processing
Number of elements		3		
Element 1		Deck		
Element 2		Superstructure		
Element 3		Substructure	Constant	Eliminated
Period of inspection/planning (years)		2		
Discount rate (%)	r	3		
Horizon (years)	T	20		
Length (m)	L_b	(5, 1800)		
Width (m)	W_b	(3, 60)		
Vertical clearance (m)	V_c	(4, 7)		
ADT (vehicles/day)	ADT	(100, 400000)		
Truck percentage (%)	p_T	(0, 0.5)		
Detour length (km)	L_d	(1,100)	Continuous	Normalized
Maintenance duration (days)	t_{ma}	(10, 60)		
Rehabilitation duration (days)	t_{rh}	(120, 240)		
Reconstruction duration (days)	t_{rc}	(300, 540)		
Traveling speed before project (km/hr)	V_a	(40, 90)		
Traveling speed during the project (km/hr)	V_b	(15, 35)		
Drift ratio in the user costs model[3]	η	(0.01, 0.1)		

Standard deviation in the user costs model	σ	(0.01, 0.1)		
Detour usage percentage (%)	p_d	(0, 0.99)		
Earthquake magnitude distribution: Lognormal mean	μ_e	(3, 5)		
Earthquake magnitude distribution: Lognormal standard deviation	σ_e	(0.01, 2)		
Earthquake occurrence rate (events/year)	ζ	(0.001, 0.1)		
Skew angle (degree)	θ	(0, 45)		
Superstructure age (years)	A_{sp}	(1, 90)		
Substructure age (years)	A_{sb}	(1, 90)		
Deck age (years)	A_d	(1, 90)		
Deck condition	CR_d	{9,8,7,6,5,4}		
Superstructure condition	CR_{sp}	{9,8,7,6,5,4}	Ordinal	Normalized
Substructure condition	CR_{sb}	{9,8,7,6,5,4}		
Number of spans	N_{sp}	(1, 60)		
HAZUS class	HZ	HWB {1, 3, 5, 8, 10, 12, 15, 17, 22}		
Soil type class	Sl	A, B, C		
Deck material	M_d	{1, 2, 3, 8}		
Design	D_t	{1,2,3,4,5,6,7,10,11, 12,14,16,21,22}	Nominal	One-hot encoding
Material	M_a	{1,2,3,4,5,6}		
Road class	RCL	Local, Major, Minor, NHS		

5-2-2 Bridge management parameters

Bridge parameters and MCS are input variables of the developed PL-IAM systems. All parameters that were used in the presented case study and their corresponding role can be found in (Asghari, Hsu and Wei, 2021). The range of these variables, that will be used for

synthesizing new assets, is based on the recorded data of Indiana bridges in the NBI. In addition, the role of each parameter in the implemented computational models, which are adopted from the consisting literature, is also provided in detail.

5-2-3 Sampling optimization results of bridges

New synthesized bridges based on the range of real bridges are developed for developing the dataset for training the ML model. Then, the MRR plan of each synthesized bridge is optimized with the GA for a 20-year horizon. The results of optimization, as well as the LCCA and bridges' parameters, are then stored for further analysis. The optimization process with the GIAMS computational core considering all sources of uncertainties associated with hazards, deterioration, and costs, takes more than 400 minutes (Asghari and Hsu, 2021). It means that synthesizing and optimizing 1300 bridges would take approximately one year. Besides, 1300 data points are far from enough for training an ML model to estimate the output of such a complex system. To bypass this obstacle, the trained DNN model developed by Asghari, *et al.* (2021a) was used to estimate the results of LCCA instead of the MCS. They showed that their developed DNN model can estimate the LCCA results with a negligible error but 5 orders of magnitudes faster than the regular MCS. Adopting the abovementioned models could reduce the optimization time to 2.4 minutes from 400 minutes. Finally, more than 1.6 million bridges were synthesized and their MRR plans were optimized individually.

5-2-4 RF training

5-2-4-1 Data preprocessing

Asset characteristics and MCS parameters, as the input parameters of ML models, should go under specific preprocessing actions before the training session. The choice of these preprocessing actions is associated with the characteristics of the datasets. Normalization,

encoding, and pruning were conducted to address the problems with the range of continuous parameters, nominal and ordinal parameters, and redundant/constant variables. Of all the asset characteristics and MCS parameters, continuous variables were normalized to have a similar range between 0 to 1. One-hot encoding was used to convert nominal parameters with k different labels into $k-1$ binary parameters. Constant parameters of asset characteristics and MCS parameters were eliminated before training to reduce the dimensionality of the problem. A summary of all the pre-processing actions for all parameters in this study is provided in Table 5.2. These pre-processing actions converted the existing features to 59 new features that were used as the input of the ML models.

5-2-4-2 Encoding MRR plans

Since the optimized MRR actions will be estimated by ML models, they will be used as the output variables. MRR actions encoding in this study is a string of numbers containing 30 values corresponding to 3 elements and 10 biannual years in the 20-year management horizon. Although the optimized MRR plans were derived as strings of binary variables, they can be encoded to decimal variables. These binary and decimal encodings are illustrated in Figure 5.5. An ensemble of ML models corresponding to each MRR action could be trained to predict the MRR actions. In the case of this study, 30 ML models that are capable of multiclass classification can be trained to estimate the decimal MRR actions. Similarly, 60 binary classification ML models (e.g., logistic regression) can be trained to estimate the binary MRR actions. These two approaches are depicted in Figure 5.6.

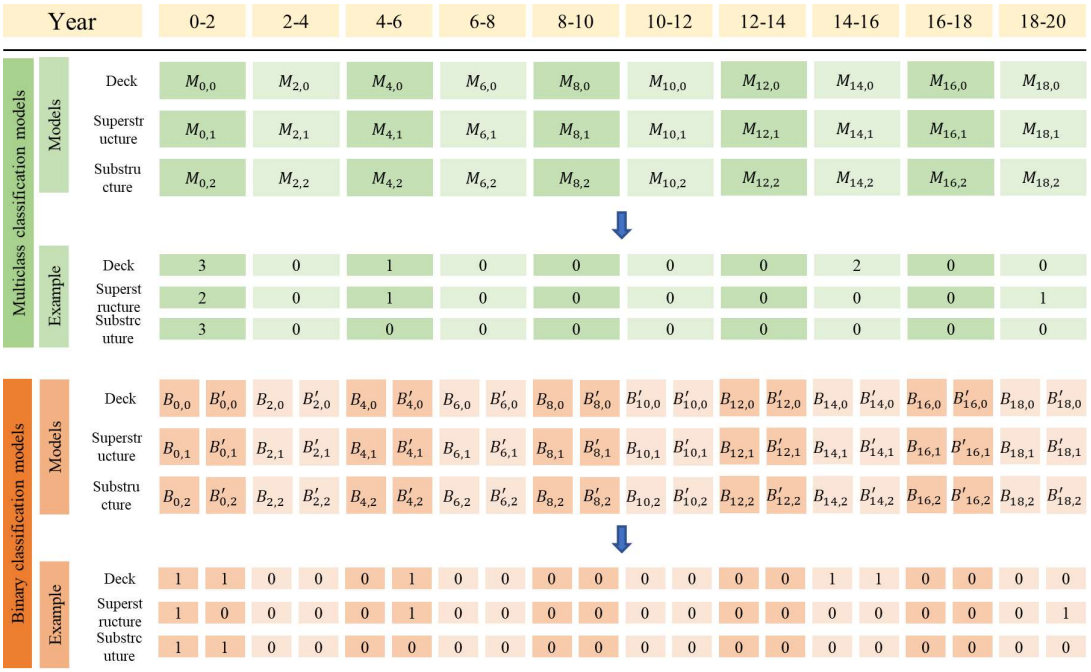


Figure 5.6. Binary and decimal encoding of MRR actions and corresponding ML approach

In this figure, $M_{i,j}$ corresponds to the multiclass classification models that predict the project (0: do nothing, 1: maintenance, 2: rehabilitation, or 3: reconstruction) for element j at year i. Also, $B_{i,j}$ and $B'_{i,j}$ correspond to the pair of binary classification models that predict the binary-encoded actions (0,0: do nothing, 0,1: maintenance, 1,0: rehabilitation, or 1,1: reconstruction) for element j at year i.

5-2-4-3 Hyperparameter tuning

The RF model consists of several hyperparameters that affect its accuracy and performance both in training and prediction. Despite the available guidelines for hyperparameter tuning of neural networks, support vector machines, and logistic regression (Ng, 2018), no specific procedure is proposed for hyperparameter tuning of DT-based models. In fact, grid search and random search are common practices in the literature for this purpose (Zhou *et al.*, 2019; R. Wang *et al.*, 2020; Campos *et al.*, 2021). In the grid search method, the prediction

accuracy of all possible combinations of hyperparameters is evaluated and the combination corresponding to the highest prediction accuracy with k-fold cross-validation is selected as the best set of hyperparameters. The search space and the hyperparameters of the RF model to be tuned are summarized in Table 5.3.

Table 5.3. Hyperparameters of RF model and the search space for tuning

Hyperparameters	**Symbol**	**Seach space**
Number of trees	N_{tr}	{100, 200, 500, 800, 1000}
Maximum depth	d	{10, 20, 50, 80, 100}
Minimum observations in leaf nodes	N_l	{1, 10, 20, 50}
Minimum samples for split	N_r	{2, 20, 40, 100}
Maximum features (total feature: N)	f^{max}	$\log_2 N$, $N^{0.5}$, N

However, this seemingly applicable approach might cause problems when the studied dataset is relatively large and has multiple outputs. Training an RF model to predict only one out of the 30 MRR actions takes approximately 20 minutes. Considering the size of the search space with 1200 unique combinations of hyperparameters, 30 MRR actions, and 5-fold cross-validation the grid search algorithm would take 2500 days. To tackle this time complexity hindrance, the grid search was conducted based on the MRR action of the first year and a randomly selected subset of the whole dataset with 10,000 data points. It assumes that the distribution of the randomly selected subset is similar to that of the original dataset.

5-3 Results and discussion

The MRR optimization of each synthesized asset with GA as the optimization module and DNN models as estimators of LCCA results takes approximately 2.4 minutes. Taking advantage of parallel processing and using an Intel(R) Xeon(R) E5-2697 CPU, 128 GB RAM, with 72 logical processors, the datapoint sampling of 1.6 million optimization results took almost 39 days. Following the hyperparameter tuning procedure proposed in the

methodology section, training each RF model on 10,000 data points, with an Intel(R) Core (TM) i7-8700T CPU @ 2.40GHz and12 Logical Processor, takes 2.1 seconds. Given the search space size provided in Table 5.3 (i.e., 1200 unique combinations of hyperparameters and 5-fold cross-validation), RF hyperparameter tuning took 1.7 hours. The set of hyperparameters of the RF model with the highest prediction accuracy is $N_{tr} = 500$, $d = 100$, $N_l = 1$, $N_r = 2$, $f^{max} = N^{0.5}$.

Finally, 30 RF models were trained on 80% of the whole data as the training set to predict the decimal form of MRR plans. To provide a benchmark for the prediction accuracy of RF, 60 logistic regression models were trained on the same dataset to predict the binary form of the MRR plans. In this process, the L2 regularization parameter was used to minimize the effect of less important input parameters and prevent overfitting. Based on the guidelines provided in (Asghari, Leung and Hsu, 2020), $L2$ regularization was set to 10^{-6}.

Two approaches were selected for validation of the results. First, the prediction performance of the trained models on the test set (i.e., the remaining 20% of the data that was not used for training) is evaluated. This is the common practice and accepted procedure in ML modeling. But the prediction performance of the trained ML models on semi-synthesized assets would not guarantee similar performance on real assets. To ensure the consistency of trained models' prediction performance on real assets, optimal MRR plans of all highway bridges in Indiana were derived using the same methodology for optimizing the semi-synthesized assets. Then, the MRR estimations of the trained models on the Indiana network were validated against the derived optimal plans. The results of these two strategies for validation of results using accuracy, precision, recall, and f-1 score are reported in Table 5.4.

Table 5.4. Prediction metrics of the trained ML models on the test set

Dataset	ML model	Accuracy	Precision	Recall	f1-score

Semi-synthesized dataset	Logistic regression	0.83	0.37	0.33	0.32
	DT	0.93	0.75	0.73	0.73
	RF	0.95	0.98	0.72	0.80
Indiana highway bridges	RF	0.89	0.85	0.79	0.86

The results of the RF models with an overall accuracy of more than 95% and an f1-score of more than 0.80 on the test set demonstrate a proof of concept for the theoretically well-founded and practically applicable methodology proposed in this study. Based on the results of Table 5.4, the prediction results of the RF model are higher than that of the logistic regression models by a large margin. In addition, the trained RF models could accurately estimate the optimal MRR plans of Indiana highway bridges with more than 4600 assets with an accuracy of more than 89% and an f1-score of more than 0.86. This implicitly implies two arguments. First, estimation of the optimization results is a highly non-linear and complex phenomenon. Therefore, linear models such as logistic regression are not capable of capturing the inherent complex relationships in this problem. Second, the RF model is an appropriate approach for predicting the optimization results as a substitute for directly obtaining the near-optimal MRR plans for the bridges with GA.

The computation time of estimation of LCCA results with MCS and the readily available DNN models as well as finding the optimization results with the GA and the trained RF models are summarized in Table 5.5.

Table 5.5. PL-AMM computation time with different methods

Methodology	**LCCA**	**Optimizer/Estimator**	**Overhead computation time**	**One asset**	**50,000 assets**
Common practice of PL-AM	MCS	GA	0	492 mins	46.5 years
Proposed in (Asghari, Hsu	DNN	GA	1 hour	2.4 mins	83.3 days

and Wei, 2021)

The proposed methodology in this study	DNN	RF	~1 month	0.01 seconds	~50 seconds

To provide a deeper and clearer insight on the enhancement of the computation time of PL-IAM, a discussion is drawn upon the Texas highway bridges with more than 50,000 bridges. Obtaining the optimal or near-optimal MRR plans for all bridges in this network with a relatively complex LCCA system (Appendix A) will take 45.6 years. However, the LCCA results and optimized MRR plans could be predicted with the trained models with acceptable prediction errors within 50 seconds. This massive improvement in computation time could only be achieved with approximately 1 month of datapoint sampling and training sessions. The improvements in its practical application in real-life problems would pay off for this one-time procedure with a relatively big overhead computational cost.

This methodology is an attempt to fill the gap between the practice and the literature on PL-IAM. This gap is mainly due to the massive computational cost of detailed project-level MRR optimization systems in which several sources of uncertainties such as hazards, costs, and deterioration are considered. Substituting MCS and GA with trained ML models, this methodology tries to estimate the results based on a semi-synthesized dataset. The prediction results provided in Table 5.4 can be regarded as proof of this concept. This methodology can be similarly applied to other asset management systems such as pavement management systems and seismic retrofit of buildings for estimating the detailed results of MCS and GA in a significantly shorter time.

Chapter 6 Multi-agent reinforcement learning for project-level intervention planning under multiple uncertainties

This chapter advances a holistic framework, from early development to final validation of results, for training RL models in IAM-related studies with more focus on the long-established aspects of IAM and less on the theoretical aspects of RL. Several sources of uncertainties, such as hazards, costs volatility, and asset deteriorations, as well as the response of assets to hazards, are simulated in a microworld based on the available and currently used models. Finally, RL agents are trained to make decisions about the time and extent of the maintenance actions on assets.

The application of RL methods in civil engineering has long been established in transportation engineering (Rasheed *et al.*, 2020). IAM domain could equally benefit from these sophisticated RL techniques. Aiming to address the limitations of the recent RL-based IAM-related studies, a holistic RL framework from early development to final validation of results is developed in this chapter. We took the MARL approach and used different off/on-policy methods (e.g., deep Q-learning and advantage actor-critic (A2C) models) to train the AI agents. Given their indisputable importance and significant role in daily lives, bridges and bridge management was selected as the example asset in the case study. The AI agents learned to select the extent and implementation time of maintenance actions on a real high-way bridge in Indiana, the US throughout a 20-year horizon. Our results showed appreciable improvement in reduction of LCC and increasing stakeholders' utilities. Among the trained agents and models, A2C models could converge to better results. The training was conducted using a microworld developed by the authors based on the previously published open-source infrastructure management platform, GIAMS (Asghari and Hsu, 2021). The RL models were also developed and trained by the currently available python libraries (i.e., Keras (Chollet,

2015) and Tensorflow (Abadi *et al.*, 2016)). The RL codes are also available at: https://github.com/vd1371/ReinforceAM.

The main contribution of this chapter to the IAM body of knowledge is, therefore, 1) proposing a framework for adding flexibility to the conventional IAM decision making, 2) developing a holistic RL framework containing the long-established components and theories of IAM practice, 3) providing a solution for taking closer to reality decisions regarding maintenance of assets. The proposed practice-oriented framework can enhance the development and extension of RL in the IAM literature and practice by both researchers and practitioners while considering various sources of uncertainties such as hazards, deterioration, and costs. Given the similarity of the RL-based decision-making process to reality, the proposed framework can potentially lead to significant improvement in costs reduction, network safety, infrastructure reliability, and more sustainable and resilient communities. Another, yet important, advantage of such an RL-based framework to the conventional MCS-HOA ones is the computational speed at the decision-making stage. Once an RL agent is trained, it can be implemented for practice during the intended time without a specific need for further training and learning. Highlighted by previous studies (Haji Hosseinloo *et al.*, 2020; Lee and Kim, 2021; Si *et al.*, 2021), the reduction in the computation time can be regarded as the fourth important and attractive characteristic of this framework alongside its flexibility in decision making and deploying complex and probabilistic models.

The remainder of this chapter is structured as follows. An overview of RL methods with some techniques for their improved learning is summarized in the next section. Next, RL development and methodology for IAM from the initial development of microworlds, constructing and training the agents, and validating RL agents' results are discussed. Then, a case study about using the proposed framework on a real bridge in Indiana, the US with comparative results is provided. Finally, a section is allocated to results and discussion.

6-1 Methodology: RL for asset management

6-1-1 Reinforcement learning method

This section provides a summary of the theoretical background of RL methodology. First, the formulation of RL and MDPs problems is discussed. Then, two major approaches, namely Q-learning and policy gradient, toward training AI agents are briefly summarized, followed by a brief enumeration of some training techniques.

The ultimate purpose of RL methods is to train an agent in a microworld for taking sequential decision making with the aim of maximizing long-run rewards. This problem can be and has been formulated as MDPs in the literature (Andriotis and Papakonstantinou, 2019; Wei, Bao and Li, 2020; Yao *et al.*, 2020; Bowes *et al.*, 2021; Ghannad, Lee and Choi, 2021). Figure 6.1 depicts a big picture of this notion in a summarized form.

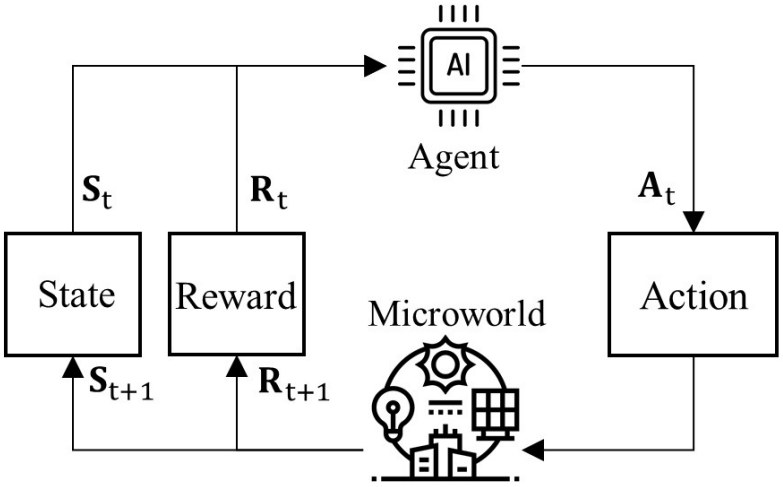


Figure 6.1. RL framework and big picture

The microworld with its discretized decision-making steps, t, has the key role in training RL agents because it provides:

1- A definition and the encoding of the (finite set of) states, **S**, for informing the agent about the microworld state $\mathbf{S}_t$ at each time step.

2- A finite set of possible actions, $\mathbf{A}$, from which the agent can select a subset of actions $\mathbf{A}_t$ at each time step.

3- A reward function, $\mathbf{R}$, that informs the agent about the consequences, $\mathbf{R_t}$, of its selected actions $\mathbf{A_t}$ at $\mathbf{S_t}$.

4- A Markovian interstate model, P_M, that determines the transition between $\mathbf{S_t}$ to $\mathbf{S_{t+1}}$ given the selected action $\mathbf{A_t}$ of the agent. (Note: if P_M could be eliminated from an MDPs problem, the whole RL algorithm could be referred to as model-free)

The selected actions by the agent based on the states are usually mapped by a policy function called π. Whether deterministic, $\pi(\mathbf{S_t}): \mathbf{S} \rightarrow \mathbf{A}$, or probabilistic, $\pi(s_t): \mathbf{S} \rightarrow P_M(\mathbf{A}|\mathbf{S}_t)$, the policy function leads to a discounted reward, with γ as the discount factor, from t through the end of the MDPs problem:

$$\mathbf{R}_t^{\pi} = \sum_{j=t}^{T} \gamma^{j-t} \mathbf{R}(\mathbf{S}_j, \mathbf{A}_j) \quad (6.1)$$

However, the stochastic nature of MDPs and this problem (e.g., stochastic transition model and even actions) make finding the expected return of rewards necessary. This expected return of rewards given all possible states and actions is usually called Q-value and can be formulated as an action-value function:

$$Q^{\pi}(\mathbf{S}_t, \mathbf{A}_t) = \mathbb{E}_{\mathbf{S},\mathbf{A}}[\mathbf{R}_t^{\pi}|\mathbf{S}_t, \mathbf{A}_t] = \mathbf{R}(\mathbf{S}_t, \mathbf{A}_t) + \gamma \mathbb{E}_{\mathbf{S},\mathbf{A}}[\mathbf{R}_{t+1}^{\pi}|\mathbf{S}_{t+1}, \mathbf{A}_{t+1}] \quad (6.2)$$

Action-value function, based on its name and definition, highly depends on the selected action. It is also essential to understand the expected return of all Q-values for all possible actions at each step t. This leads to a value function:

$$V^{\pi}(\mathbf{S}_t) = \mathbb{E}_{\mathbf{A}_t}[\mathbf{R}_t^{\pi}|\mathbf{S}_t, \mathbf{A}_t] = \mathbb{E}_{\mathbf{A}_t}[Q^{\pi}(\mathbf{S}_t, \mathbf{A}_t)] \quad (6.3)$$

As the ultimate purpose of the RL agents is to maximize the long-run rewards, at each step t, the agent is expected to select the action with maximum value. Although linear programming

can be used for maximizing the value function in simple problems, value or policy iteration methods are usually used for real-life and complex problems.

Before continuing and explaining these two major approaches in the following sub-sections, a brief discussion on the partially observable MDPs (POMDPs) is required. Previous studies (Andriotis and Papakonstantinou, 2019, 2021; Murugesan *et al.*, 2020) assumed that the exact state of the microworld cannot be observed and took this assumption into account. Acknowledging the validity of this assumption, the authors intend to highlight that this assumption seems not to be considered in the IAM practice. The IAM agencies and decision-makers collect data regarding the state of the assets periodically and conduct decisions accordingly (e.g., in (FHWA (Federal Highway Administration), 2012)).

6-1-1-1 Value iteration

As was discussed in the previous section, the agent can learn (find the action with corresponding maximum value function) through iterative interactions with the microworld. After each interaction, the Q-value can be updated by the difference between the previously perceived Q-value and new results, e, with a learning rate α for deterministic policies with discrete action spaces:

$$Q^{\pi^{new}}(\mathbf{S}_t, \mathbf{A}_t) = Q^{\pi^{old}}(\mathbf{S}_t, \mathbf{A}_t) + \alpha e \quad (6.4)$$

Two approaches, namely off-policy (e.g., Q-learning (Watkins and Dayan, 1992)) and on-policy (e.g., SARSA (Rummery and Niranjan, 1994)), have been taken for calculating this error. The on-policy approach uses the selected action Q-value at $t+1$, Eq. (6.5), whereas the off-policy approach uses the optimal Q-value at $t+1$, Eq. (6.6):

$$e = \mathbf{R}(\mathbf{S}_t, \mathbf{A}_t) + \gamma Q(\mathbf{S}_{t+1}, \mathbf{A}_{t+1}) - Q(\mathbf{S}_t, \mathbf{A}_t) \quad (6.5)$$

$$e = \mathbf{R}(\mathbf{S}_t, \mathbf{A}_t) + \gamma \max_{\mathbf{A}_{t+1} \in \mathbf{A}_{t+1}} Q(\mathbf{S}_{t+1}, \mathbf{A}_{t+1}) - Q(\mathbf{S}_t, \mathbf{A}_t) \quad (6.6)$$

This iterative approach can be feasibly performed through tabular learning. However, this approach is infeasible for problems with continuous state spaces. Past studies have proposed

using function approximators to estimate the Q-values for each action given a vectorized and encoded form of the state $\mathbf{S}$. Linear function using gradient descent (Sutton and Barto, 2018) and neural networks (Mnih *et al.*, 2013) are two of the well-known function approximators. Further details can be found in the cited references.

6-1-1-2 Policy iteration

When the action space of a problem is continuous or the policy definition is probabilistic, value iteration techniques cannot be used. Instead, another family of algorithms each of which is a variation of the policy gradient theorem (Sutton *et al.*, 1999) can be used to directly estimate the actions. Deep policy gradient methods are an example of such algorithms. These algorithms directly train a policy function $\pi(\mathbf{S})$ by:

$$g^{\pi} = \mathbb{E}_{\mathbf{S},\mathbf{A}}[\sum_{t} \nabla^{\pi} log\pi(\mathbf{A}_t|\mathbf{S}_t) Q^{\pi}(\mathbf{S}_t, \mathbf{A}_t)] \quad (6.7)$$

Since $Q^{\pi}(\mathbf{S_t}, \mathbf{A_t})$ in Eq. (6.7) has a non-zero mean, which leads to instability and lowers convergence speed during the optimization (policy iteration). To tackle this issue, using an advantage function, ψ^{π}, with expected mean equal to zero has been suggested and used by previous studies (Sutton and Barto, 2018):

$$g^{\pi} = \mathbb{E}_{\mathbf{S},\mathbf{A}}[\sum_{t} \nabla^{\pi} log\pi(\mathbf{A}_t|\mathbf{S}_t)\, \psi^{\pi}(\mathbf{S}_t, \mathbf{A}_t)] \quad (6.8)$$

The advantage function can take a variety of forms, but a common form is deducting the value function from the action-value function (Konda and Tsitsiklis, 2001):

$$A^{\pi}(\mathbf{S_t}, \mathbf{A_t}) = Q^{\pi}(\mathbf{S_t}, \mathbf{A_t}) - V^{\pi}(\mathbf{S_t}) \quad (6.9)$$

Function approximation can be used to estimate the results of the policy function, $\pi(\mathbf{A}|\mathbf{S})$, (the actor model) and the value function $V(\mathbf{S})$ (the critic model). These two can also be combined in a single neural network with shared weights and layers and form an A2C model. Further details can be found in (Silver *et al.*, 2014).

6-1-1-3 Training techniques

Regardless of the type of the RL algorithm, the pseudo-code for training RL agents can be summarized in the following algorithm:

Algorithm 6.1. The basic training algorithm of RL agents

1: Initialize the model Θ
2: **for** episode **in** (1, max_episode):
3: **for** t **in** (1, T):
4: **Select** best action A_t given $\mathbf{S}_t$ using Θ
5: **Act** based on $\mathbf{A}_t$, calculate $\mathbf{R}_t$, and find $\mathbf{S}_{t+1}$
6: **Update** Θ based on $\mathbf{S_t}, A_t, \mathbf{R_t}, \mathbf{S_{t+1}}, A_{t+1}$
7: **return** Θ

However, several techniques have been proposed in the literature to improve the convergence speed, reduce instability, and reach better models. Some of the most well-known techniques in this regard are explained in the following.

The initialized model in Line 1 of Algorithm 6.1 can exploit its randomly initialized knowledge and yield some strategies at the beginning of the training session. Consequently, the model can be updated and converge to a point with these exploited strategies. But this approach can make the agent biased toward the randomly initialized model's decision. A common practice to tackle this issue is to select the actions A_t both by randomly exploring all possible actions and greedily exploiting the decisions made by the model. ϵ-greedy is an example for such practices (Sutton and Barto, 2018). ϵ-greedy algorithm selects random actions with probability of ϵ and optimal actions using the Θ model with probability of $1 - \epsilon$. Also, the initial value of ϵ decays as time goes by, agents experience several episodes of the microworld and learn different strategies. Exploration rate decay and its initial values are hyperparameters and could vary in different problems given the specific requirements and

settings of each problem. The exploration of RL agents with probabilistic policies takes the form of randomly selecting the actions based on their probabilities instead of merely selecting the action with the highest probability.

The model update in DQN is referred to as the updating of weights and biases of the neural networks. If a model that is being updated repeatedly is also used for calculating the Q-values, the agents will experience severe instability during the learning process. One common technique to overcome this issue is to develop two models: 1) an online model that is being updated after each episode, 2) a target model that finds the best action at each step and yields Q-values for updating the online model. Notably, the target model's network weights are replaced by the online model's network weight after a user-defined number of episodes. Further details can be found in (Mnih *et al.*, 2013).

Hasselt (2010) showed that the conventional update of the Q-learning estimators can be biased leading to more required iterations for learning and less accurate policies. The solution proposed in that study was to use two estimators Q^{π^A} and Q^{π^B} and update one estimator with the predicted Q-values of the other one. Later, (Van Hasselt, Guez and Silver, 2015) showed that the benefits of double Q-learning such as reaching less biased Q-values and more accurate policies also hold for DQN.

The exploitation and exploration experiences of agents during the training session can be stored and used for further learning. In fact, there is no reason that the previously experienced outcomes of agents' interaction with the microworld be of less importance than the new ones. In addition, using the stored experiences would take negligible time in comparison to interacting with the microworld and generating new experiences. As a result, experience replay (also called memory replay) has been consistently used in the RL literature and practice to speed up the learning process (Yang *et al.*, 2015; Brandi *et al.*, 2020; Ye *et al.*, 2021). In another study, Schaul et al. (2016) claimed vanilla sampling from the stored

experiences is inefficient and samples with higher impacts on the learning process should be sampled with higher probability. They introduced importance-sampling weight for each sample based on the temporal difference error. Details of this process can be found in (Schaul *et al.*, 2016).

6-1-1-4 Multi-agent RL (MARL)

Multiple agents, instead of one, could interact with a microworld in different settings. Cooperative, competitive, and mixed are some of these settings (Zhang, Yang and Başar, 2021). In cooperative settings, agents share a common goal and reward. In competitive settings, agents compete with each other in a zero-sum game. Mixed setting refers to games in which each agent has its own interest that could be in line with or against other agents. Cooperative or mixed settings, shown in Figure 6.2, mostly resemble the managerial decision-making in IAM with different assets or elements.

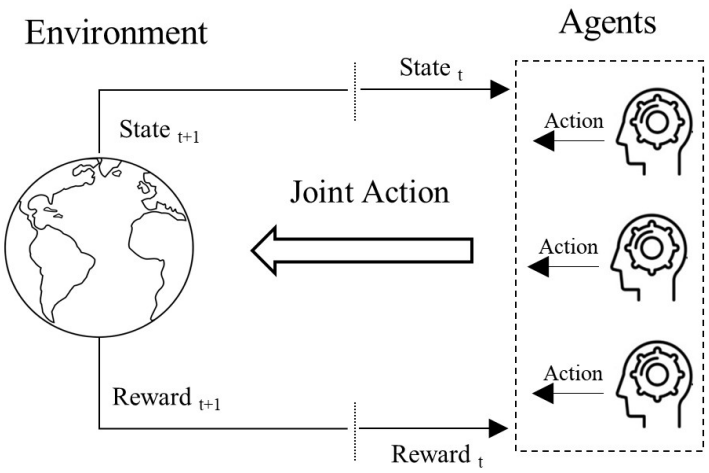


Figure 6.2. MARL abstract form

Despite a few limitations of this approach such as dimensionality expansion, its effectiveness and practicality have already been demonstrated in previous studies (Nguyen, Nguyen and Nahavandi, 2020; Zhang, Yang and Başar, 2021) including the IAM related ones (Andriotis and Papakonstantinou, 2019).

This section discusses a high-level and intuitive yet practical approach toward developing IAM systems, microworlds, training RL models, and finally validating the results of the trained RL models. As the first step, an IAM system based on the settings of researchers and practitioners' problems and inherent characteristics of assets should be developed. To name a few, hazard models, deterioration patterns, different types of costs, financial discount rates, and stakeholders' utilities are some of these settings and characteristics. Then, a higher-level model, a microworld, should be developed on top of the developed IAM to provide the foundation for training RL algorithms. Next, RL models should be designed and trained. Details of such a process are provided later. Finally, the results of the trained RL model should be properly validated. A summary of the proposed framework is illustrated in Figure 6.3.

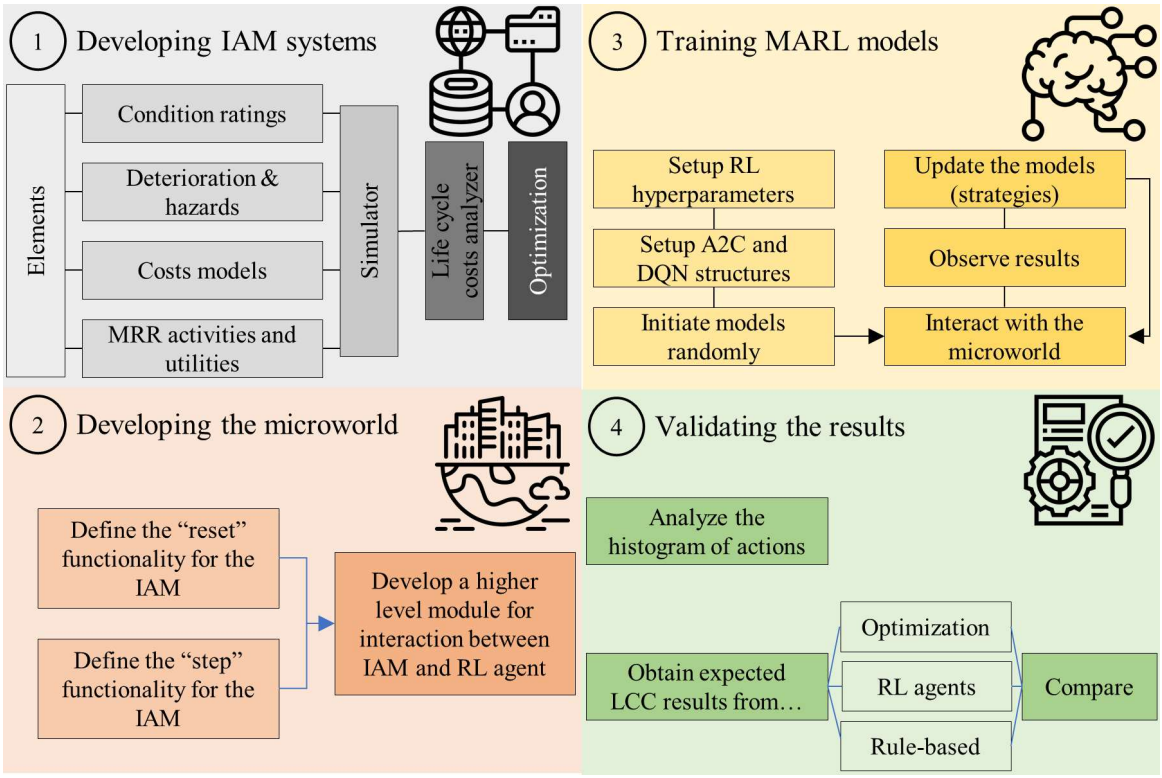


Figure 6.3. A summary of the proposed methodology for RL in IAM

6-1-2 Developing IAM systems

IAM systems development domain has a rich history with a multitude of studies consistently trying to improve the previously developed systems. The improvement in these systems could be in terms of implementing more accurate models describing assets (Yang and Frangopol, 2019; David Y Yang and Frangopol, 2020), more advanced optimization techniques finding more profitable results (Mondoro, Frangopol and Liu, 2018; Yang, Frangopol and Teng, 2019), or more sophisticated methodologies reducing the computation time with the focus on real-life practicality (Yang, Hsieh and Kung, 2012; Asghari, Hsu and Wei, 2021). Asset management and its development for different systems have unique characteristics and require full-length studies, or sometimes books, to cover every single important detail. However, most IAM systems' structures share several different modules and models. Following the work of Asghari and Hsu (2021), we briefly summarize the main parts of PL-IAM systems in Figure 6.4.

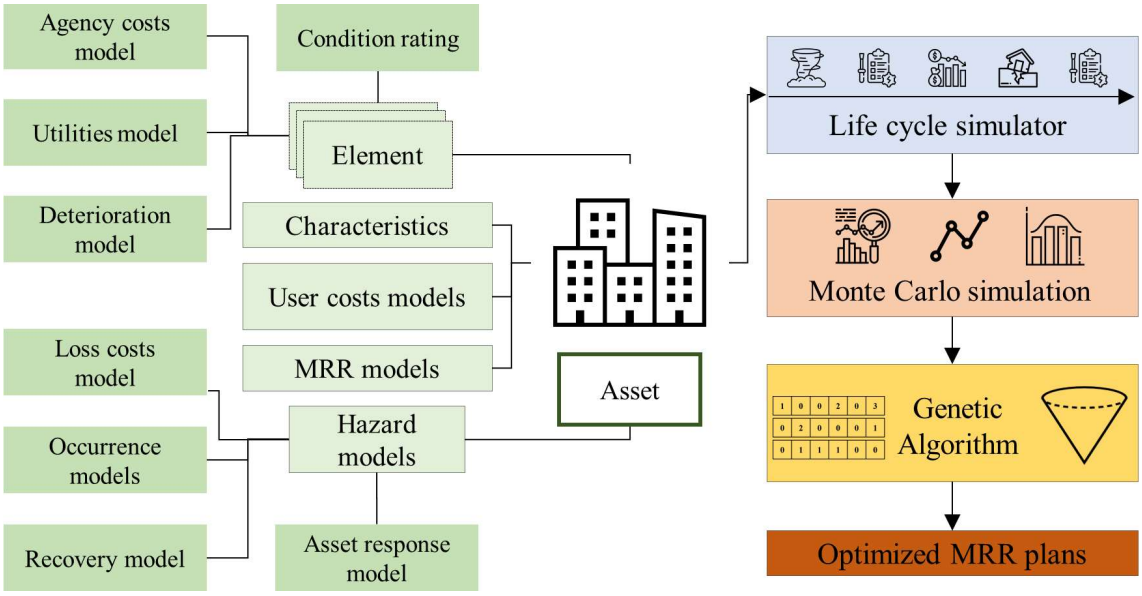


Figure 6.4. The structure of PL-IAM systems

The first, and probably the most important, step in any IAM system is the CR definition. The CR is a measure to describe the state of assets in comparison to their newly built condition. Different types of CR measures for different assets currently exist in the literature for different assets (Jeong *et al.*, 2018; Peraka and Biligiri, 2020). The condition of assets is constantly under the influence of different environmental agents (e.g., aging, loading, and chemicals) and hazards (e.g., earthquakes, hurricanes, floods). As the second step in developing an IAM system, deterioration models and hazard models should be meticulously implemented based on reliable guidelines (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003) or previous studies (Sinha *et al.*, 2009). The changes in the condition of assets, whether due to deterioration or response to hazards, are associated with different costs. Traffic delay, loss of lives, injuries, and structural damage to a part of assets are some examples. Defining these costs and their projections in the future is the third step of any IAM system and should be clearly defined and modeled. Preventive actions, such as regular maintenance, rehabilitation, or even reconstruction (MRR), can minimize unfavorable consequences of deteriorative agents and catastrophic events. These actions have three major characteristics in IAM. First, these actions lead to significant direct and, more importantly, indirect costs. Direct costs refer to the actual dollars spent by agencies or responsible bodies for performing preventive actions. Indirect costs, however, are burdened by the community members. The increase in travel time and excessive fuel consumption are some examples of these indirect costs. Second, the effectiveness of preventive actions in improving the condition of assets depends on various parameters such as the previous conditions or the quality of materials. Third, different stakeholders (i.e., users and agencies) have different perspectives on preventive actions. Regardless, the utility of preventive actions has been one of the major decision-making

parameters in previous studies. Defining preventive actions with their roles is the fourth main step in IAM development.

The ultimate purpose of PL-IAM is to optimize the long-run and LCC and utilities. Each project-level IAM system contains a module for simulating unprecedented events and phenomena. Depending on the complexity and type of the models used in shaping the IAM system, deterministic simulations or the MCS could become useful in this regard. Finally, an optimization algorithm depending on the problem characteristic is usually developed and designed on top of the LCC analyzer. The optimization algorithm finally returns a long-term plan of the extent and type of preventive actions in the life cycle of an asset. The generated long-term plan is expected to result in the optimal combination of costs and utilities among all possible long-term sets of actions.

6-1-3 Developing the microworld

The main role of a microworld in training RL models is to simulate all the possible aspects of a problem (IAM in our case) that affect, and are affected by, the decisions of an agent. Therefore, a microworld structure naturally shares several components with the general structure of IAM systems. However, microworlds should contain a few more specific RL-related tools that will be utilized during the RL training procedure. First, microworlds should contain a “step” function that makes appropriate changes to the simulated IAM system given a certain set of actions. RL training is usually conducted in an iterative manner. As a result, microworlds should also contain a “reset” function that brings all states of an IAM system to their initial state once the simulation in one life cycle reaches its end. Both functions are expected to yield information about the state of the IAM system and the consequences of actions decided by the agent. This information must be properly encoded for use in neural networks models. The details of such an encoding process are highly dependent on the properties of each problem, but they usually should abide by the conventional feature

encodings in the ML domain such as one-hot encoding and normalization. The abstract structure of a microworld is depicted in Figure 6.5 to provide a graphical abstract of the described idea.

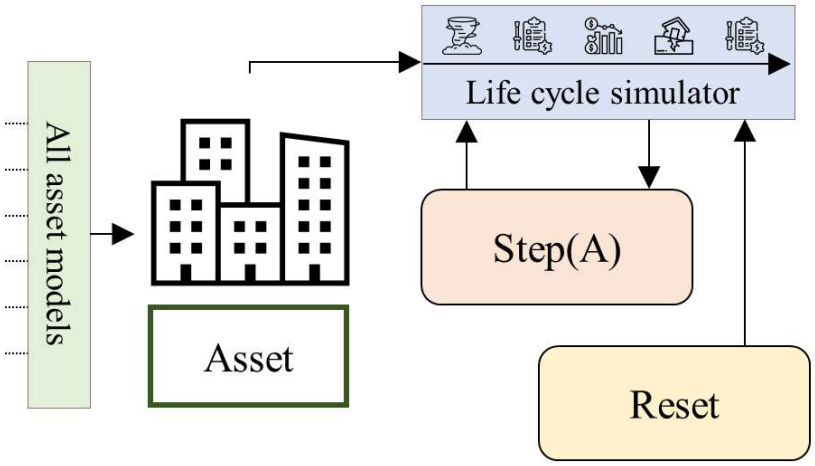


Figure 6.5. The abstract structure of a microworld based on a regular IAM framework

6-1-4 Training MARL models

The essence of RL training as well as some training techniques were discussed in the previous section. Following the discussion on the IAM microworlds and RL training, we provide a pseudo-code of two commonly used algorithms (i.e., DQN and A2C) in the following. IAM systems usually make a decision about several elements of an asset. The objective function that is used in previous studies usually aggregates information (e.g., utilities and costs) about all elements into one single function. This approach has been called the multi-attribute utility function method (Bai *et al.*, 2013) and mostly resembles the cooperative setting for MARL agents. Algorithm 6.2 and Algorithm 6.3 provide the pseudo code for MARL training in PL-IAM with DQN and A2C approaches.

Algorithm 6.2. Pseudo-code for MARL for PL-IAM: DQN approach

1: Initialize the models $\Theta_1 : \Theta_n$ with random weights //Initialize one model for each element

2: Initialize the target models $\Theta_1': \Theta_n'$ with random weights //Initialize one target model for each element

3: Initialize memory buffer $\beta_{M_1}: \beta_{M_n}$ //Initialize one memory buffer corresponding to each model

4: Initialize the IAM microworld ϕ_M

5: **for** episode **in** (1, max_episode): //Value iteration for a maximum number of episodes

6: $\mathbf{S}_t$, done $\longleftarrow \phi_M$.reset() //Reset the microworld and get the initial state of the asset

7: **for** t **in** (1, T): //Iterate over decision making steps in management horizon

8: $A_t \longleftarrow$ with probability ϵ select A_t randomly from $\mathbf{A}_t$ //Exploration with epsilon greedy among all possible actions

9: $A_t \longleftarrow$ otherwise, select optimal A_t //Exploiting the models

10: **if** double_DQN: using $\Theta_1: \Theta_n$ //Actions are selected from models in double DQN

11: **else:** using $\Theta_1': \Theta_n'$ //Actions are selected from target models in regular DQN

12: $\mathbf{R}_t, \mathbf{S}_{t+1} \longleftarrow \phi_M$.step($A_t$) //Take one step using the selected action, calculate the rewards, and find the next state

13: **for each** element j:

14: **store** encoded($\mathbf{S}_{j,t}, A_{j,t}, \mathbf{R}_{j,t}, \mathbf{S}_{j+1,t}$) in β_{M_j} //Storing the states, actions, and rewards

15: **if** double_DQN: select A_{t+1} using $\Theta_1: \Theta_n$ //Finding maximum Q-values of next step

16: **else:** select A_{t+1} using $\Theta_1': \Theta_n'$

17: $Q_{t+1}^{\pi^{old}} \longleftarrow$ Q-values of A_{t+1}using $\Theta_1': \Theta_n'$

18: $Q_t^{\pi^{new}} = \mathbf{R}_t + \gamma Q_{t+1}^{\pi^{old}}$ //Updating Q-values

19: **for each** element j:

20: **update** Θ_j using $\mathbf{S}_{j,t}, A_{j,t}, Q_{j,t}^{\pi^{new}}$

21: every now and then: $\epsilon = \epsilon \times \lambda_\epsilon$ //Using exponential decay for epsilon-greedy algorithm

22: every now and then: update $\Theta_1 : \Theta_n$ with $\Theta_1' : \Theta_n'$ //Update the models using the target models

23: every now and then: //Memory replay

24: **for** k **in** (1, max_epochs):

25: **update** $\Theta_1 : \Theta_n$using samples from $\beta_{M_1} : \beta_{M_n}$

26: **return** $\Theta_1 : \Theta_n$

Algorithm 6.3. Pseudo-code for MARL for PL-IAM: A2C approach

1: Initialize the models $\Theta_1 : \Theta_n$ with random weights //Initialize one model for each element

2: Initialize memory buffer $\beta_{M_1} : \beta_{M_n}$ //Initialize one memory buffer corresponding to each model

3: Initialize the IAM microworld ϕ_M

4: **for** episode **in** (1, max_episode): //Value iteration for a maximum number of episodes

5: $\mathbf{S}_t$, done $\longleftarrow \phi_M$.reset() //Reset the microworld and get the initial state of the asset

6: **for** t **in** (1, T): //Iterate over decision making steps in management horizon

7: $A_t \longleftarrow$ select A_t from $\pi(\mathbf{S}_t)$ //Select actions from the actor models with their corresponding probability

$C_t^\pi \longleftarrow$ find C_t^π from $V(\mathbf{S}_t)$ //Find the critic values using the critic models

8: $\mathbf{R}_t, \mathbf{S}_{t+1} \longleftarrow \phi_M$.step($A_t$) //Take one step using the selected action, calculate the rewards, and find the next state

9: **for each** element j:

10: **store** encoded($\mathbf{S}_{j,t}, A_{j,t}, \mathbf{R}_{j,t}, \mathbf{C}_t, \mathbf{S}_{j+1,t}$) in β_{M_j} //Storing the states, actions, and rewards

11: $\mathbf{R}' \leftarrow$ discounted $\mathbf{R}$ // Discounting rewards

12: $\mathbf{A_d^{\pi}} = \mathbf{R}' - \mathbf{C}^{\pi}$ // Finding advantages

13: **update** actor models of $\Theta_1 : \Theta_n$ using $(\mathbf{S}, \mathbf{A}, \mathbf{A_d^{\pi}})$ //Updating actor models

14: **update** critic models of $\Theta_1 : \Theta_n$ using $(\mathbf{S}, \mathbf{R}')$ //Updating critic models

15: **return** $\Theta_1 : \Theta_n$

Both these algorithms contain several hyperparameters. Exploration rate, exploration decay rate, buffer size for memory replay, frequency of updating models, frequency of updating exploration rate, frequency of memory replay, discount rate, hyperparameters regarding the structure of the neural networks model (layers, activation functions, learning rate, etc.) are examples of such hyperparameters. Unfortunately, there is no solid method for tuning these hyperparameters before training. As a matter of fact, hyperparameter tuning of RL models is a craft of experience and a trial-error procedure. That said, some of these hyperparameters, such as discount rate, learning rate, and reward function, have higher impacts on the learning of the models than other ones. The discount rate implies the effect of previous experiences from the beginning of each episode (t = 0) up to t. $\gamma = 0$ means zero impact from previous decisions and states. $\gamma = 1$ means a complete influence from previous decisions and states. Learning rate, as in other gradient descent-based learning models, has a significant impact on the convergence point and speed of RL models. But perhaps, the reward function, which yields positive rewards for taking desired strategies and negative rewards for taking unfavorable decisions, has the highest impact on the learning process. Although some guidelines have been provided in the literature (Dewey, 2014), the reward function, also, should be designed by the researchers and practitioners.

6-1-5 Validating the results

The results of trained RL models, similar to all other analytical models, should be validated. If properly designed, RL models could potentially converge to a point and yield some strategies. However, the usefulness of these strategies must be compared with that of the conventional methods. Two approaches, namely analyzing the strategies with expert judgment and comparing results with other methods, have been separately adopted in the literature for validating the results of the trained RL models. We emphasize both these approaches and explain them in the following. First, the histogram of actions in the life cycle of assets should be drawn graphically (Andriotis and Papakonstantinou, 2019). Then, the derived strategies should be validated by expert judgments to avoid infeasible or incorrect solutions. Moving the pawn like the bishop in the game of chess is an example of an incorrect solution. Annual maintenance of a near collapse bridge for 50 years is another example in the context of IAM. Second, the expected outcomes of the RL models should be compared with that of other methods such as rule-based strategies (Yang *et al.*, 2015; Chemingui, Gastli and Ellabban, 2020) or optimization algorithms (Krishna Lakshmanan *et al.*, 2020; Si *et al.*, 2021). One important aspect of comparing the results of trained RL models with that of optimization algorithms is the consistency between the hyperparameters and optimization settings. Obviously, this comparison is of less use if the objective function or constraints of an optimization algorithm is different from the reward function of trained RL models. The same holds for some other hyperparameters such as discount rate.

6-2 Case study: RL for project-level bridge management

Bridges play an important role in the transportation network of each community. Due to their indisputably important role and their degraded states around the world (e.g., (American Society of Civil Engineers, 2021)), BMSs have been the focus of hundreds of studies so far

(Alysson, M. and Liang, 2018). We also use the BMS as a showcase of the discussed concept in this chapter. The following paragraphs start by discussing a high-level and big picture of the BMS implemented in this chapter. Then, a summary of the implemented microworld will be briefed followed by a report on the feature encoding and hyperparameters of the MARL model.

6-2-1 The BMS overview

The BMS used in this chapter is readily available in an open-source project called GIAMS (Asghari and Hsu, 2020). Its developers started this project with the aim of facilitating the research and development in IAM using up-to-date programming languages in an open-collaboration setting. GIAMS currently contains a fully developed BMS based on the Indiana, US, highway bridges with models based on previous studies and guidelines (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003; Li and Sinha, 2004; McGuire, 2004; Sinha *et al.*, 2009; FHWA (Federal Highway Administration), 2012; TexasDOT, 2020). It focuses on long-term preventive maintenance intervention planning for deck, superstructure, and substructure of bridges. Details and in-depth explanations of this BMS can be found in (Asghari and Hsu, 2021). This BMS has also been used in previous studies chapters. Providing the details of all models used in this BMS would lengthen this chapter. To keep this chapter concise and to the point, we summarized the models used in this BMS with their original references in Table 6.1.

Table 6.1. Summary of the adopted BMS models

No.	Model	Reference
1	Condition rating	(FHWA (Federal Highway Administration), 2012)
2	Deterioration models	(Sinha *et al.*, 2009)
3	MRR models and agency costs	(Sinha *et al.*, 2009)

4	Hazard generation models	(McGuire, 2004; USGS, 2020)
5	Hazard response and cost models	(FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003)
6	User costs models	(TexasDOT, 2020)
7	Utilities	(Li and Sinha, 2004)
8	Life cycle cost analyzer and optimization	(Asghari and Hsu, 2021)

6-2-2 The microworld development

Following the guidelines provided in the previous section, we developed a microworld on top of the GIAMS's BMS named IndianaEnv. Similar to other microworlds (sometimes called games, environments), IndianaEnv can create a virtual simulation environment to interact with designed RL agents. It, first, loads all the characteristics of the bridge with Id = 10 from the national bridge inventory of Indiana, US. Then, all corresponding models such as hazard models, cost models, and utility models are added to shape a holistic simulation environment. IndianaEnv design follows the common practice and architecture of widely used microworlds developed by OpenAI (Brockman *et al.*, 2016). It contains a "step" function that receives a set of actions for all elements of the bridge and performs necessary simulative changes on the microworld. Then, it returns the consequences of the selected actions as well as the next state of the microworld. Its "reset" function also reverts all changes made in the microworld to their initial states. The code for this microworld is readily available online (Asghari and Hsu, 2020).

6-2-3 MARL structure

The proposed structure for the RL agents deciding about the preventive actions in the BMS is illustrated in Figure 6.6.

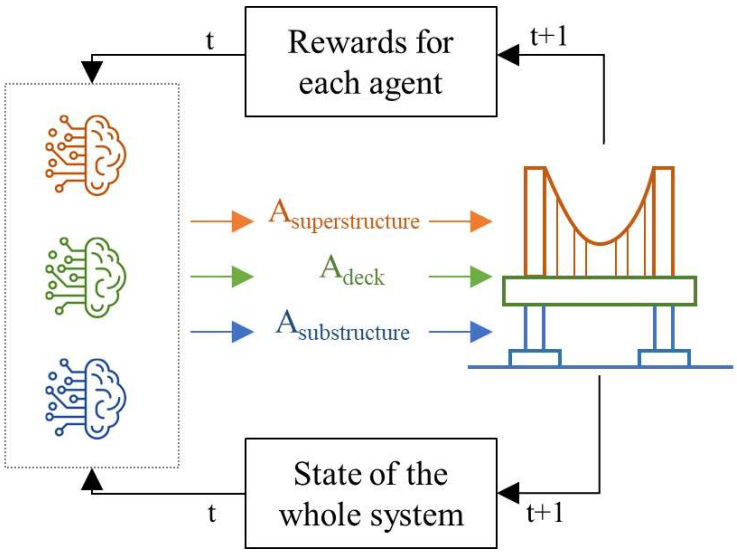


Figure 6.6. MARL structure for the BMS

Whether DQN models or A2C models, each element of the studied bridge has a responsible agent. Given the microworld's state, each agent takes a decision (do nothing, regular maintenance, rehabilitation, reconstruction) separately. The whole or a part of the consequences of agents' decisions are shared among them. The details of the states used as the input for the agents, feature encodings, and hyperparameters are provided in the following.

6-2-4 Feature encoding and hyperparameters

A subset of the features/characteristics of a bridge (or an asset in general) are considered constant during its life cycle. The length, width, and the number of understudied elements and their materials are some examples. Intuitively, these features do not affect the decision-making process directly as they do not change in time. Their effect, however, can be seen inside other models such as deterioration, costs, and perhaps utility models.

Other variable characteristics being associated with the sources of uncertainties heavily affect the outcomes of the decision-making process. Three sources of uncertainty are considered in this case study: deterioration, earthquake, and costs. Out of all characteristics of

a bridge, elements' CR and their age are related to the deterioration model (Note: The Markovian deterioration model in this study is adopted from (Sinha *et al.*, 2009) and is a function of age, element type, element material, and previous condition of elements). Variables related to the earthquake phenomenon, such as the time and magnitude of the last occurred earthquake, are theoretically irrelevant to this problem. This is mostly due to the uncertain nature of earthquakes occurrence. In other words, the history of earthquakes cannot lead any decision-maker to the exact occurrence time and magnitude of unprecedented earthquakes. That said, the impact of earthquakes has been directly and indirectly considered in the overall costs and utilities of actions. Costs and their uncertain changes have been modeled using the Wiener process with drift. Further details on this model and its implementation can be found in Chapter 3. The effect of this third source of uncertainty on decision-making has been considered as a feature equal to the percentage of deviation from the expected and previously projected costs. The remaining budget and the time of decision-making are other features that have been considered effective parameters in the RL modeling. All the features with their range and encoding are summarized in Table 6.2.

Table 6.2. Summary of features (states) and their encoding

State	Description	Range	Encoding
S_1^*	Deck condition	0-9	$(S_1^* - 4.5)/4.5$
S_2^*	Deck age	0-40	$(S_2^* - 20^*)/20$
S_3^*	Superstructure condition	0-9	$(S_3^* - 4.5)/4.5$
S_4^*	Superstructure age	0-40	$(S_4^* - 20)/20$
S_5^*	Substructure condition	0-9	$(S_5^* - 4.5)/4.5$
S_6^*	Substructure age	0-40	$(S_6^* - 20)/20$
S_7^*	Remaining budget (%)	0-1	N/A
S_8^*	Decision step	0-20	$(S_8^* - 10)/10$

S_9^*	Deviation of costs	-0.1-0.1	N/A

*: The median age of the Indiana network of bridges is 20 years

The objective function used in (Asghari and Hsu, 2021), Eq. (6.10), is used as the guideline for developing the reward function:

$$R = \frac{\sum_j U_j}{\left(\sum_j C_{A_j} + C_U\right)^{0.2}} \quad (6.10)$$

Where U = utilities of actions, C_A = agency costs, C_U = user costs, and j = element index. This function aggregates all utilities and costs and yields one final value. But we used this function as a guideline and define a new reward function with a slight change:

$$R_j = \begin{cases} \dfrac{U_j}{\left(C_{A_j} + C_U\right)^{0.2}}, & \text{if enough budget and valid action} \\ -5 \quad , & \text{else} \end{cases} \quad (6.11)$$

The details of the constraint (i.e., budget and valid actions) can be found in (Asghari and Hsu, 2021). If the agent, however, takes an illegal action, futile action, or surpasses the available budget, the reward will be equal to -5. This negative reward was set with try and error. An example of the reward engineering process with hypothetical numbers is depicted in Figure 6.7. Actions in this figure, 0: do nothing, 1: maintenance, 2: rehabilitation, 3: reconstruction.

Year	0	2	4	6	8	10	12	14	16	18
Action	3	0	2	2	0	2	0	1	0	0
Reward	8	0	0	1	0	1	0	1	0	0
Budget limit	0	0	0	0	0	0	0	-5	-5	-5
Futile action	0	0	-5	0	0	0	0	0	0	0
MRR constraint	0	0	0	-5	-5	-5	-5	-5	-5	-5
Final reward	8	0	-5	-5	-5	-5	5	-5	-5	-5

Figure 6.7. Reward engineering summary

The authors acknowledge that the summative assumption of non-linear functions does not hold. But this approach was taken due to the following reasons:

1- The agency costs (C_{A_j}) are usually much smaller than the user costs (C_U). Past studies speculated that the ratio of agency costs to user costs can be even 1:10 (David Y. Yang and Frangopol, 2020b). Given that the user costs are equal for each agent corresponding to each element, the summation will be basically on the utilities with a relatively large common denominator. Therefore, the difference between the outcomes of Eq. (6.10) and the sum of reward functions, Eq. (6.11), for all elements will be negligible.

2- If the agents share a common reward function, the learning process will take longer, and the agents might not even learn. An example is provided to explain this issue in more detail. In an imaginary case, maintenance of the bridge deck is assumed to be the best strategy with enough available budget. At the same time, the agent responsible for the newly built superstructure decides to reconstruct it (sub-optimal strategy) and the agent responsible for the substructure chooses to do nothing (the optimal strategy). Since the superstructure agent's decision has led to a costly decision beyond the available budget, the reward will be negative. If this reward is shared among all agents, we are punishing the deck and substructure agents for taking the optimal strategies. This means hundreds and thousands of explorations by agents are wasted by being punished most of the time for taking the right decisions.

Nevertheless, the outcomes that are used for comparing the expected long-run utilities and rewards of the MARL models and baselines are similar for the sake of consistency. This important issue was discussed in Section 6-1-5. Not to mention, other objectives such as energy consumption and traffic-related carbon emissions can be similarly used as the objective function and the reward function. Not to mention, different reward functions would lead to different strategies. Also, it is possible to use different reward engineering approaches for obtaining different strategies. An example of these approaches is penalizing the agent if

the condition rating of an asset becomes worse than a certain threshold. These ideas require further and in-depth studies and are out of the scope of this dissertation.

The hyperparameters are set by trial and error and given the recommendations in previous studies (Ng, 2018; Santos, Ferreira and Flintsch, 2019; Asghari, Leung and Hsu, 2020; Yao et al., 2020; Ghannad, Lee and Choi, 2021). The list of hyperparameters for the DQN and A2C network as well as the MARL hyperparameters are summarized in Table 6.3 and Table 6.4.

Table 6.3. DQN and A2C networks hyperparameters

Model	**Hyperparameter**	**Value**
Both	Type of layers	Fully connected layers
	Input layer activation function	*tanh*
	Hidden layers activation function	*relu*
	Optimizer	Adam
	Hidden layers	30, 30
	Regularization type and value	*L2*, 0.000001
	Learning rate	0.001
DQN	Output layer activation function	*linear*
	Cost function	MSE (Mean of Squared Error)
	Output layer	4
A2C	Output layer activation function	*Softmax* (actor) + *Linear* (critic)
	Cost function	Categorical cross entropy (actor) + MSE (critic)
	Output layer	4 (actor) + 1 (critic)

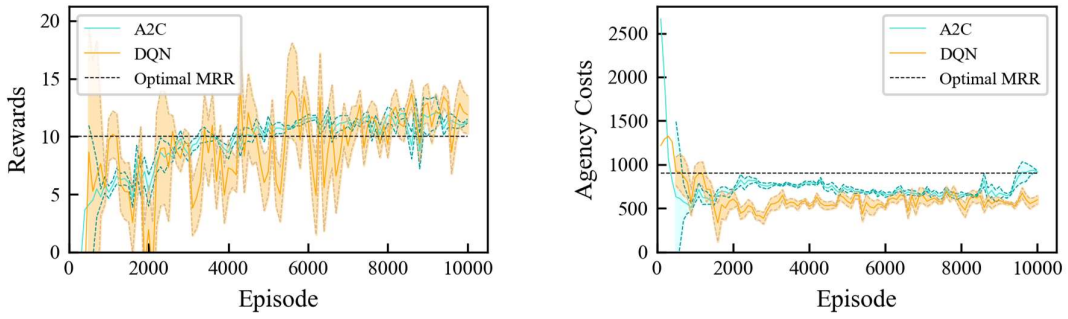
Table 6.4. MARL hyperparameters

Model	**Hyperparameter**	**Value**
Both	Discount factor (γ)	0.97

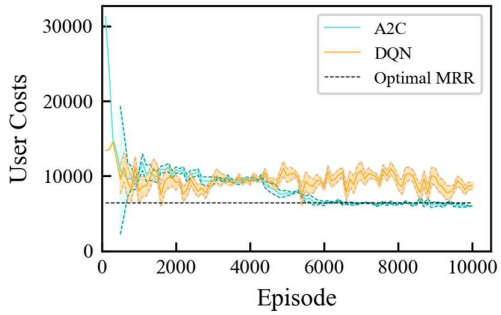
	Maximum number of episodes	10000
DQN	Exploration probability (ϵ)	0.5
	Exploration decay factor (λ_ϵ)	0.001
	Updating ϵ frequency (episodes)	5
	Exploration decay function	$\epsilon/(1+\lambda_\epsilon \times \frac{episode}{10})$
	Buffer size	10000
	Epochs	10
	Batch size	1000
	Updating target model frequency	100 episodes

6-3 Results and discussion

The proposed MARL models were trained using an Intel(R) Core (TM) i7-8700T CPU @ 2.40GHz and12 Logical Processors. Notably, the same machine was used for deriving the optimal MRR plans. The following paragraphs discuss the outcomes of the proposed MARL framework through a comparative analysis.



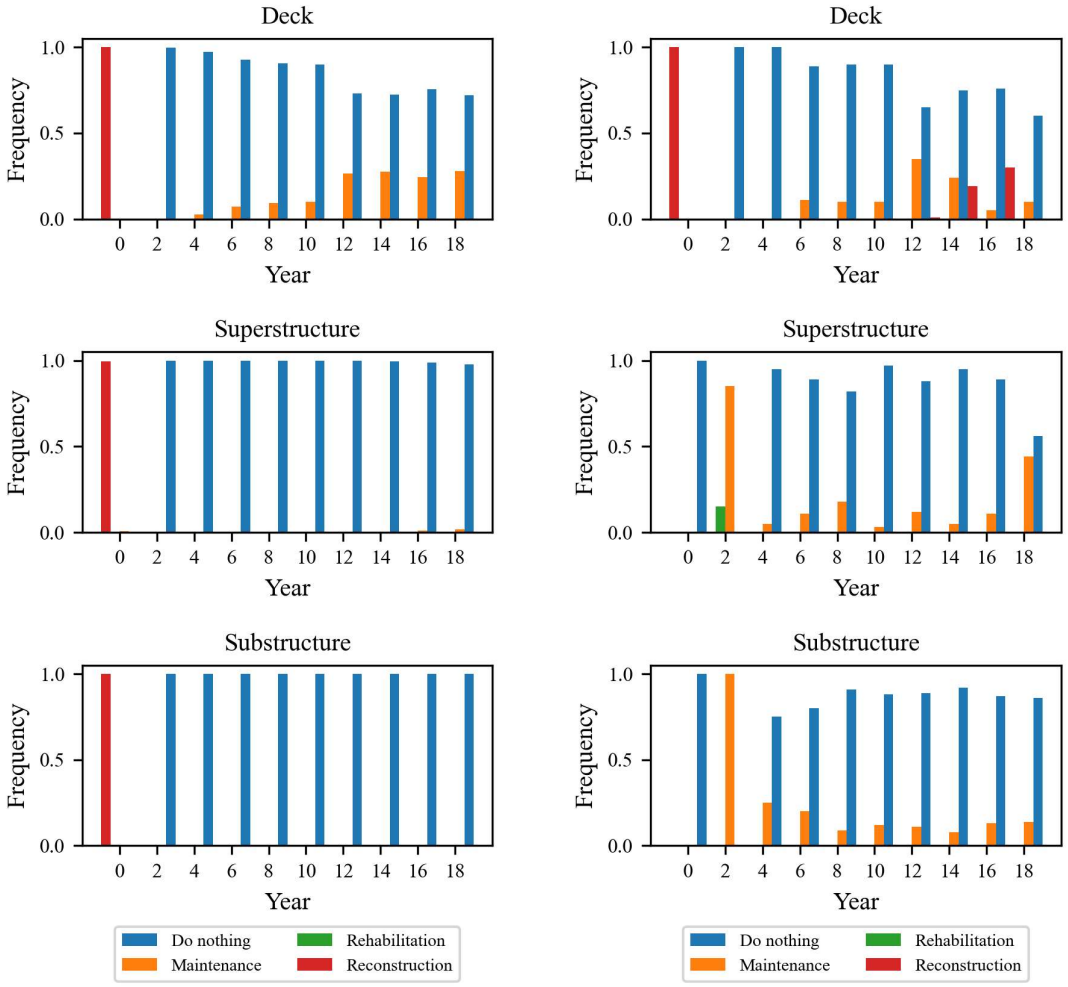
(a) (b)



(c)

Figure 6.8. Expected (a) rewards, (b) agency costs, (c) user costs convergence

Figure 6.8 illustrates the convergence of the rewards and costs during the learning process. Both DQN and A2C models could converge to higher expected rewards in comparison to the optimal MRR plans (Note: The expected values are derived by simulating the LCC with the trained agents for 1000 rounds). In fact, A2C and DQN models, in comparison to the optimal MRR plan could improve the rewards by up to 14% in the management horizon. The trained models took different policies which are associated with different expected costs (see Figure 6.9). The A2C models could reduce agency costs and user costs by approximately 7% and 5% respectively. DQN was more successful in reducing the agency costs (35% reduction) but at the expense of a considerable increase in user costs (31%). The difference between the selected strategies is depicted in Figure 6.9.

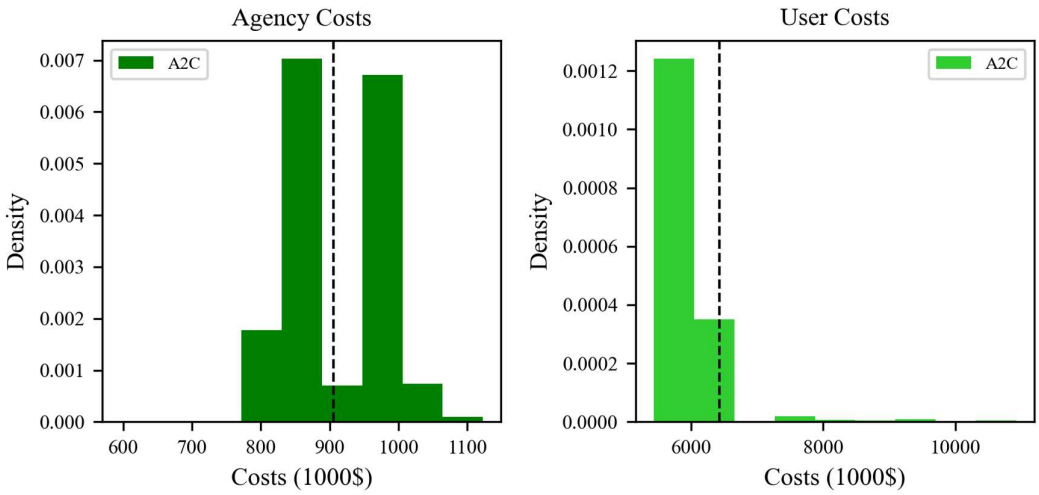


(a) (b)

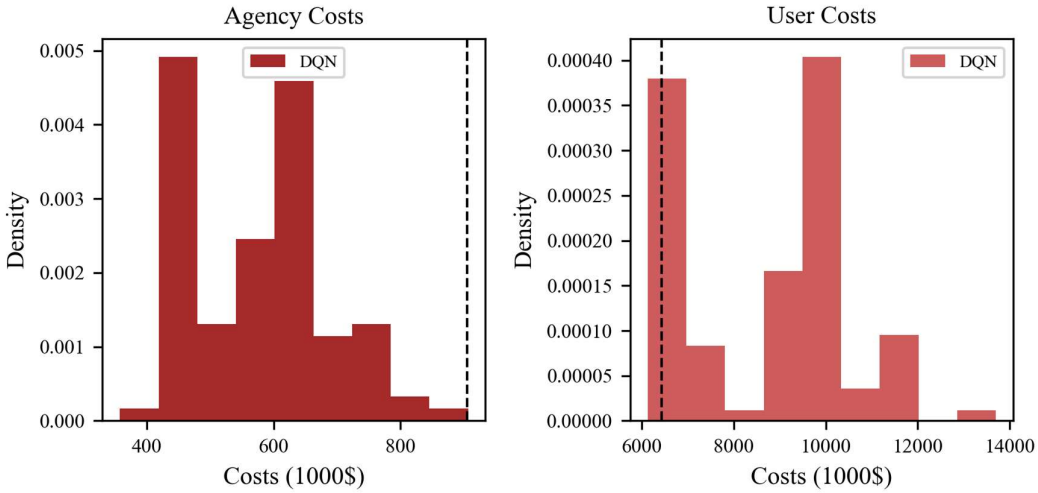
Figure 6.9. Histogram of action in (a) A2C and (b) DQN models

Both A2C and DQN agents that were responsible for the deck took a similar strategy which leads to an initial reconstruction of the deck with almost no future action. But these models seem not to have reached a consensus on the superstructure and substructure. The A2C agents took a safer approach leading to the reconstruction of both superstructure and substructure with almost no future actions. But the DQN agents had decided to keep maintenance of these elements in the long run. The difference between these two models could be stemmed from the different initialization of the neural networks models or the learning rate in the learning

process. Nevertheless, this difference can be beneficial as they provide different options for the decision-makers. Once a set of hyperparameters are tuned for the learning process, asset managers and decision-makers could select between the trained agents with different strategies based on their expert judgment. However, adding more constraints and limitations to the microworld can potentially force the agents to take similar policies during the training.



(a)



(b)

Figure 6.10. Histogram of agency and user costs for (a) A2C, and (b) DQN

Figure 6.10 depicts the histogram of costs for the two approaches. The comparative analysis based on these figures reveals that the agency costs as a result of the A2C models' strategies are less than that of the optimal MRR value with a probability of 55%. But the user costs associated with this set of strategies are expected to be less than the optimal MRR plan's user costs with a 97% probability. These probabilities for the DQN model are almost %100 and 10%. In other words, the strategies that are taken by the DQN agents most often lead to fewer agency costs and more user costs than the optimal MRR plan.

The MCS-HOA and RL methods' computational complexity both at the training and decision-making stages vary significantly. Using the mentioned computational machines, both DQN and A2C agents were trained in 42 minutes while the optimal MRR plans were reportedly achieved within 492 minutes. It should be noted that the same IAM system (analytical code) and the same machine were used for all these models. While the optimal MRR plans and the trained agents can assist the decision-makers in a negligible time, the MCS-HOA framework lacks complexity in decision making. It means that the whole optimization session should be conducted one more time when new observations are available. That said, HOA algorithms are relatively simpler in terms of development and hyperparameter tuning in comparison to the RL method leading to less time spent during the development session.

Chapter 7 Network-level infrastructure asset management with multi-agent actor-critic reinforcement learning

The major objective of this chapter is to develop an RL-based framework for NL-IAM under multiple uncertainties and various constraints. This chapter is a stepping stone for further extension of the RL mindset to NL-IAM. It is in line with Chapter 6 of this dissertation which focused on PL-IAM. NL-IAM being formulated as an MDPs problem, in comparison to PL-IAM, is inherently associated with a theoretical challenge: the curse of dimensionality. These obstacles make the training of RL agents computationally infeasible or even impossible. Part of the aim of this chapter is also to address these challenges. We proposed using a statistical summary of the network instead of using detailed information on the state of all assets and elements to tackle the curse of dimensionality. Given the advantages of actor-critic (A2C) models, such as stability in learning and faster convergence, the framework was put forward with a decentralized multi-agent advantage actor-critic (DMA2C) approach. The applicability of such a framework was showcased using a subset of highway bridges in Indiana, US network. In detail, the DMA2C models were trained to take intervention actions (do nothing, maintenance, rehabilitation, and reconstruction) for 48 bridges through a 20-year management horizon. These actions and decisions should have been made in face of stochastic deterioration patterns, unprecedented hazards, and fluctuating costs. The expected results of the trained agents significantly outperformed the baseline strategy derived by IUC (Patidar, Labi, Morin, Paul D. Thompson, *et al.*, 2011) optimization and optimal life-cycle actions (Asghari, Hsu and Wei, 2021). The RL agents were trained on the GIAMS environment (Asghari and Hsu, 2021) and constructed with the Python programming language with common libraries Keras (Chollet, 2015) and Tensorflow (Abadi *et al.*, 2016).

The code of the whole chapter is openly available at https://github.com/vd1371/ReinforceAM.

This chapter mainly contributes to the CEM body of knowledge by constructing the first RL-based NL-IAM decision-making system. The proposed methodology is superior to the conventional methods in multiple ways. First, it can provide long-term network-level optimal intervention plannings to decision-makers without overly simplifying the decision-making process. Second, the RL agents incorporate flexibility in face of uncertainties in the decision-making process instead of deriving rigid plans. Third, our results showed that this methodology can potentially lead to more favorable results in terms of stakeholders' utilities and costs. In summary, the proposed framework can assist decision-makers and asset managers with taking decisions that are close to reality leading to less incurred costs and higher stakeholders' utilities. These improvements can take a part in global efforts toward sustainability and the resiliency of communities. Although training the DMA2C agents in this framework took place in a considerable time, the decision-making by these agents is conducted in a negligible time. In other words, once the models are trained, they can serve for the whole life cycle without the need for further modifications. The solution provided for addressing the curse of dimensionality is equally important among the contributions of this chapter. These solutions are not limited to CEM application nor NL-IAM analysis and can be similarly applied to other MDPs problems with extremely large state and action space.

The next section of this chapter provides details of the proposed methodology from early development to implementation details and validation are discussed. This section is followed by a case study on a selected subset of bridges in Indiana, the US. Finally, the results and discussion are discussed in detail.

7-1 Methodology

In this section, the proposed methodology for NL-IAM using DMA2C models is explained. First, the IAM microworld (environment) with which the agents interact and learn is discussed on a high level. Then, the challenges in training DMA2C models for a network are addressed. Finally, validation of the proposed strategies by the trained DMA2C models is briefly explained. The developed and trained DMA2C models are capable of yielding optimal strategy given the uncertain state of the assets in a network and the environment. This methodology can theoretically outperform the traditional methods in terms of the expected life-cycle costs and stakeholders' utilities. This statement is also supported by the results of the case study.

7-1-1 The microworld: IAM System

A microworld in the RL framework mainly plays the role of simulating all occurrences and aspects of a decision-making system. By observing the state of the microworld, agents interact (i.e., take actions) with the microworld and receive deterministic or probabilistic rewards. In the context of IAM, a microworld means assets and all phenomena that can affect the state of assets in their life cycle. To further explain, the state of assets is usually altered by 1) deterioration due to environmental agents, aging, and fatigue, 2) response to hazards such as hurricanes, earthquakes, and floods. Meanwhile, asset managers can take improvement actions and improve the state of assets to enhance network safety, reduce risks in face of unprecedented hazards, and improve the serviceability of the infrastructure assets in communities. Although the improvement actions render considerable costs to the responsible bodies, they have some utilities. Inspired by the utility theory (Fishburn, 1968) the utilities demonstrate favorability of consequences of actions. IAM's body of research is mature and full of comprehensive studies concentrating on improving IAM systems in a variety of ways

such as using more accurate descriptive models, more accurate predictive models, or more efficient optimization algorithms. This study stands on the recent enhancements and achievements of this gigantic line of research by using previously developed IAM systems. A full description of developing asset management frameworks and microworlds is omitted to keep this chapter concise and to the point. For more details, please refer to Chapter 3 and previous studies (Frangopol, 2011; Asghari and Hsu, 2021). An IAM microworld is essentially an IAM system that can interact with an RL agent by taking actions at each step of the life-cycle simulation and returning the consequences of the undertaken actions.

7-1-2 Training DMA2C models

A2C models are common algorithms and their training procedure can be readily found in the literature (Konda and Tsitsiklis, 2001; Sutton and Barto, 2018). Therefore, we skip discussing the basic training procedure and focus on the challenges in DMA2C models (i.e., the curse of dimensionality) instead. Finally, the pseudo-code for training DMA2C models for NL-IAM is provided.

The state and action space size in the NL-IAM problem is a function of the number of assets and their elements in a network. To further explain, if i actions could possibly be selected for each asset with j elements at each time step, a network with n assets will have $i^{j \times n}$ different possible sets of actions. The same holds for the state space. If the state representation of each element contains k variables, the state-space size of a network with n assets will be $k \times j \times n$. This issue can be tackled by assigning a unique RL decision-making agent to each element of each asset. Consequently, the action space of each agent will become significantly smaller equal to i^j. However, the responsible agent cannot take proper decisions while ignoring the state of other elements and assets in its residing network. On the other hand, informing agents about every single detail of the network will not resolve the

curse of dimensionality for agents. We, however, propose using a statistical summary (e.g., average, 25 percentile, 75 percentile, maximum, minimum) of the states of all assets and elements in a network. As a result, the state space size of an element (of an asset) with k variables and m statistical summary representatives will be $k + mk$.

7-1-3 Validating the results

Validation of the results of RL models is one necessary step in RL training and development. Supported by the literature (Andriotis and Papakonstantinou, 2019, 2021), two approaches are recommended for validating the results of the trained RL models including DMA2C. First, the validity of proposed strategies should be manually judged and examined by experts in the field to prevent any incorrect or infeasible strategies. Second, the expected life-cycle results such as agency costs, user costs, and stakeholders' utilities should be compared with a baseline. Achieving this baseline using the common optimization framework of IAM studies (i.e., MCS) for modeling uncertainties and heuristic optimization (HOA) methods is a viable and practical approach. However, this approach might not be equally applicable for large networks of assets because of computational time. Rule-based strategies have already been employed in previous studies (Yang *et al.*, 2015; Wei, Wang and Zhu, 2017; Park *et al.*, 2019; Zhang *et al.*, 2019; Chemingui, Gastli and Ellabban, 2020; Zou, Yu and Ergan, 2020; Gupta *et al.*, 2021; Lissa *et al.*, 2021), can be used instead of optimized results. Nevertheless, we have proposed and utilized an optimization strategy based on the outcomes of a series of previous studies in this chapter. The details of this optimization strategy will be provided in the following sections.

7-2 Case study

The economy, safety, and almost every aspect of life in modern communities heavily rely on transportation networks. But, transportation networks components such as bridges are in

critical condition (American Society of Civil Engineers, 2021). In the US, for example, this concern has led to the passing of a tremendously comprehensive upgrade and maintenance of the transportation network (Cochrane, 2021). Given their undeniable important role in the transportation network, bridges have been the subject of hundreds of IAM studies. Due to these reasons, we selected a BMS to showcase the application of DMA2C in this area. In addition, the readily available microworld based on the IBMS helped us focus on extending the application of RL and DMA2C rather than starting from scratch (i.e., developing another microworld). In the following sections, first, a summary of the employed microworld based on a subset of Indiana, US highway network of bridges is provided. Then, we explain the structure and details of training DMA2C models such as a statistical summary for summarizing the network and reward engineering. Finally, a baseline for validating the results of the trained models using life-cycle optimized plans and the incremental utility-cost ratio is provided.

7-2-1 BMS and the microworld

BMSs try to inform decision-makers and asset managers about long-term optimal intervention and improvement strategies regarding the elements of bridges. Bridges have multiple elements (FHWA (Federal Highway Administration), 2012) one or some of which can be the focus of BMS models. Superstructure, substructure, and deck are some of the major elements that were selected for analysis in this study. Different measurements are used to objectively describe the condition of these elements (Jeong *et al.*, 2018). This study takes advantage of the CR proposed by the Federal Highway Administration of US (FHWA (Federal Highway Administration), 2012) which is also the prevailing measurement in the NBI of the US. Being affected by environmental factors (e.g., radiation and chemicals), the condition of these elements is altered in time. Not to mention, fatigue and aging are other contributors to the deterioration of assets and their elements. Stochastic (e.g., Markovian

models) models have been already proposed in the literature to model the uncertain phenomenon of deterioration (Thompson *et al.*, 1998; Frangopol, 2011). Being inspired by the IBMS (Sinha *et al.*, 2009), this study also used the proposed deterioration models used in IBMS. In addition to being affected by the ever-lasting deterioration phenomenon, assets undergo hazardous and catastrophic events in their lifetime. These hazards have adverse effects on the condition of elements in an unprecedented fashion. Following the previous works in this line of research and Chapter 6, this study also considered earthquake and seismic analysis as the main source of hazards. Modeling earthquakes and asset response to them is a common practice and has been well documented in the literature. Followingly, this study used the history of earthquakes available in USGS (USGS, 2020) and models suggested by FEMA-NIBS (FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences, 2003) and McGuire (McGuire, 2004). Corrective and improvement actions have different extents and subsequent consequences. Followed by previous studies (Sinha *et al.*, 2009; Bai *et al.*, 2013), maintenance, rehabilitation, and reconstruction as well as do nothing have been the main types of intervention actions used in this study. These actions can ensure the safety of assets and also strengthen the asset in face of hazards leading to enhancing resiliency measures but at a considerable cost. The first category of costs are the costs associated with maintenance and rehabilitation actions, such as materials and manpower, and are usually burdened by the responsible asset managers and decision-making bodies. These costs, which are usually called agency costs, are modeled by the equations and guidelines provided in (Sinha *et al.*, 2009). The intervention actions render a less visible, yet gigantic amount of, set of costs that are burdened by community members with excessive fuel consumption and travel time delay due to the unavailability of under maintenance assets. These costs have also been meticulously modeled and documented in the

literature. This study used the models suggested by the Texas department of transportation (TexasDOT, 2020). Apart from the costs, the favorability of actions should be and has been, modeled using the utility theory (Li and Sinha, 2004). These models and formulas are all already incorporated in an open-access project developed by authors called GIAMS (Asghari and Hsu, 2021). Going through the details of all models and formulas requires a full-length study. All details are already available in a relatively recent published study (Asghari and Hsu, 2021) and are also available online (Asghari and Hsu, 2020). This BMS has recently undergone development and contains an RL environment (microworld) which has been used in Chapter 6.

Given the existing computational limitations, a subset of bridges in the Indiana network was selected to showcase the applicability of the framework. This subset of bridges consists of all bridges with estimated future average daily traffic (ADT) of more than 100,000 and with the age of more than 25 years old. Although this is not the whole network of Indiana bridges, it still consists of 48 bridges and can be considered a relatively large problem. A discussion on the extensibility and applicability of the proposed framework is provided in the results and discussion sections.

7-2-2 DMA2C structure and training

The DMA2C models comprise a unique agent responsible for taking improvement actions for each element of an asset in a network. Each A2C agent is a neural network-based model with a special structure. That is, the same set of input and hidden layers are used for the estimation of actions and critical values. Similarly, the same network is used for training and updating weights by observing new actions and critical values. Further details regarding the neural network structure of A2C could be found in (Mnih *et al.*, 2016). The DMA2C models for a

network with n assets and each asset with j elements will have $n \times j$ responsible and decision-making A2C agents (Figure 7.1).

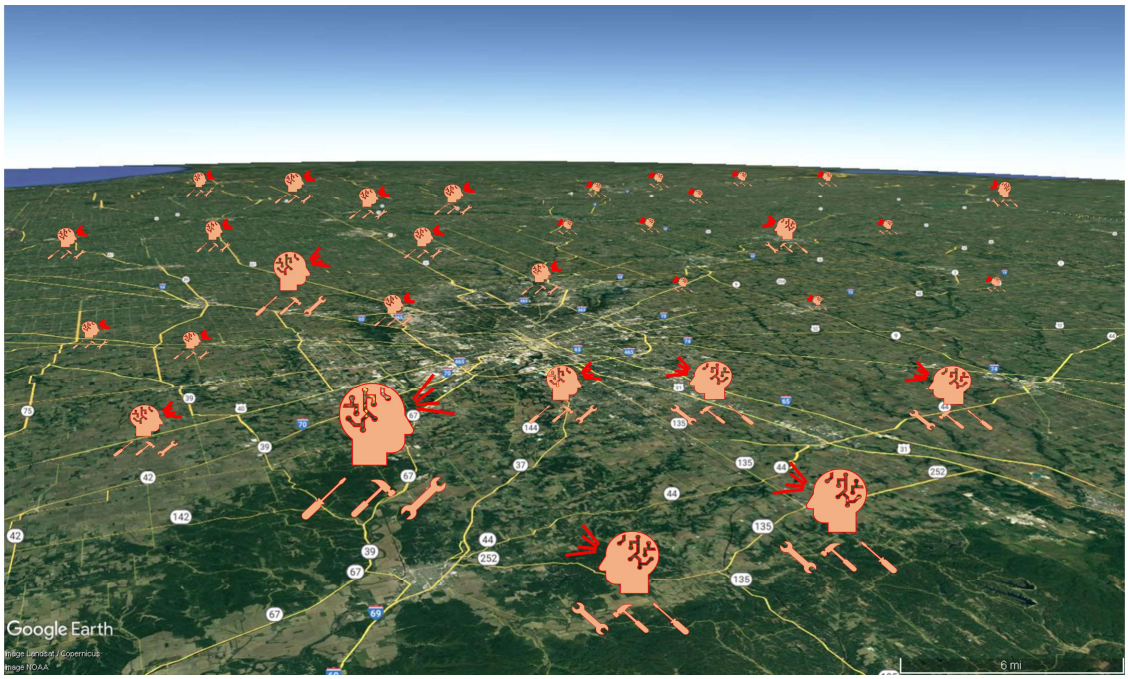


Figure 7.1. A schematic DMA2C structure for a network of assets

7-2-2-1 Feature engineering

The input vector for each agent is a combination of private information about its responsible asset and shared information about the network. The private information about the state of the asset comprises the variables including the encoded condition and age of all elements of a bridge (i.e., superstructure, substructure, and deck), the deviation of the user costs from the projected trend, and the decision making time in the management horizon. This deviation is not only a function of the prices but also the future ADT for each asset. The future ADT for each bridge can be estimated but variations could drastically change the outcome of the decision making. A full discussion on these features is available in Chapter 6. The shared information based on the statistical summary of the network is a part of the novelty of this study. Being shared among all agents in the network, the statistical summary informs the agents about the overall state of the network instead of informing them about every single

detail of all assets and elements in the network. This statistical summary includes average, standard deviation, minimum, maximum, median, 25, and 75 percentiles of all elements' ages and conditions as well as the user costs deviation for each asset. The feature engineering for the DMA2C model is summarized in Table 7.1.

Table 7.1. Feature engineering summary

	Features	Description
Private information	$S_1^*,\ S_2^*, S_3^*$	Deck, superstructure, substructure condition
	S_4^*, S_5^*, S_6^*	Deck, superstructure, substructure age
	S_7^*	Deviation of user costs
	S_8^*	Step
Network information	$S_9^*, \dots, S_{15}^*$	Ages of all decks in the network statistical summary
	$S_{16}^*, \dots, S_{22}^*$	Ages of all superstructures in the network statistical summary
	$S_{23}^*, \dots, S_{29}^*$	Ages of all substructures in the network statistical summary
	$S_{30}^*, \dots, S_{36}^*$	Conditions of all decks in the network statistical summary
	$S_{37}^*, \dots, S_{42}^*$	Conditions of all superstructures in the network statistical summary
	$S_{43}^*, \dots, S_{49}^*$	Conditions of all substructures in the network statistical summary
	$S_{50}^*, \dots, S_{56}^*$	Deviations of all assets' user costs statistical summary

7-2-2-2 Reward engineering

Reward engineering and a sound definition of reward function are one of the most important steps in training RL models. Unfortunately, there is no guideline about the definition of rewards. Therefore, reward functions should be defined heuristically and in a trial and error fashion. However, the agents learn exactly what they are being trained for. Following the previous project-level study of authors and the objective function used in deriving the baselines, we used the same reward function for each asset:

$$R = \frac{\sum_j U_j}{\left(\sum_j C_{A_j} + C_U\right)^{0.2}} \quad (7.1)$$

Where U, C_A, and C_U are the utility, agency costs, and user costs of the actions associated with element j. A discussion on the additivity of non-linear objective function was put forward in previous studies (Bai *et al.*, 2013). Considering this issue is kept for future studies. The proposed definition of reward would encourage the agents to take intervention actions all the time as there is no consequence for their futile or costly actions. However, agencies and responsible bodies' decisions are most often limited by their monetary resources. The budget constraint should be and has been considered in the models as well. Lagrangian relaxation or domain-specific manual penalization for addressing constraints is proposed in the literature (Andriotis and Papakonstantinou, 2021). Theoretically well-founded Lagrangian relaxation, however, might not be easily fit to all variations of constraints in the IAM problem. Taking advantage of the manual penalization, this study implemented two different types of penalization for strategies beyond the budget limitations and futile actions. These two penalizations are different in nature and, therefore, different approaches are required for addressing them in the context of DMA2C modeling. The subtle difference in these two approaches arises from the fact that an agent should not be penalized for inappropriate choices nor be rewarded by the optimal choices of other agents. Simply put, each agent is rewarded for its intervention actions and penalized for its futile strategies. However, all agents that were responsible for breaching the budget constraints are penalized. A schematic overview of the reward and penalty engineering in this study for two assets each with one element is depicted in Table 7.2. It should be noted that the penalty value in this figure is derived by trial and error. The total reward for element j regarding all actions of assets i is calculated with:

$$R_{total_j} = \frac{1}{N}\sum_i \sum_j (R_{ij} + P_{b_{ij}}) + (R_j + P_{f_j}) \quad (7.2)$$

Where P_b= penalty for exceeding the available budgets and P_f= penalty for futile actions.

Table 7.2. Reward and penalty engineering overview

Year	0	2	4	6	8	10	12	14	16	18
Asset 1 condition before	6	6	9	9	9	9	9	9	9	8
Actions* for asset 1	0	3	0	1	0	2	0	0	0	1
Asset 1 condition after	6	9	9	9	9	9	9	9	8	9
Asset 2 condition before	4	9	9	9	9	8	7	6	6	5
Actions for asset 2	3	0	0	3	0	0	0	1	0	2
Asset 2 condition after	9	9	9	9	9	7	6	6	5	7
Rewards for asset 1	0	14	0	0	0	0	0	0	0	3
Rewards for asset 2	16	0	0	0	0	0	0	0	0	5
Futile actions penalties for asset 1	0	0	0	-10	0	0	0	0	0	0
Futile actions penalties for asset 2	0	0	0	-10	0	0	0	-3	0	0
Not enough annual budget penalty	0	0	0	-10	0	0	0	0	0	-10
Total rewards asset 1	8	21	0	-15	0	0	0	0	0	2
Total rewards asset 2	24	7	0	-15	0	0	0	-3	0	4

*0: do nothing, 1: maintenance, 2: rehabilitation, 3: reconstruction

7-2-2-3 Hyperparameters and hyperparameter tuning

Two different sets of hyperparameters should be separately tuned in training RL models: the hyperparameters of the approximator model and the hyperparameters of the RL method. Setting these hyperparameters is both a craft of experience and a trial and error procedure. However, good starting points for starting the trial and error could be found in the literature (Ng, 2018; Santos, Ferreira and Flintsch, 2019; Asghari, Leung and Hsu, 2020; Yao *et al.*, 2020; Ghannad, Lee and Choi, 2021). The used hyperparameters in this study are summarized in Table 7.3.

Table 7.3. DMA2C neural networks and RL hyperparameters

Hyperparameter	Value

A2C Models	Type of layers	Fully connected layers
	Input layer activation function	tanh
	Hidden layers activation function	relu
	Optimizer	Adam
	Hidden layers	80, 80
	Regularization type and value	L2, 0.000001
	Learning rate	0.0001
	Output layer activation function	Softmax (actor) + Linear (critic)
	Cost function	Categorical cross entropy (actor) + MSE (critic)
	Output layer	4 (actor) + 1 (critic)
RL	Discount factor (γ)	0.97
	Max number of episodes	30000

Some hyperparameters are fixed given the theory and problem formulation. For example, the cost function and output layer activation functions are fixed given the theory of the A2C models. Some hyperparameters have relatively less important and do not need to be accurately tuned. Following the recommendations in the literature (Ng, 2018; Asghari, Leung and Hsu, 2020), the number of hidden layers and nodes should be adequately, but not grotesquely, large to capture the non-linearity of the problem. Then, the type and value of the regularization parameter should be tuned to prevent overfitting in the models. Learning rate is perhaps the most important hyperparameter that determines the final outcome of the learning process. If a large number is set for the learning rate, the agents quickly converge to a sub-optimal point. Consequently, less frequent visited spaces in the search space will never be visited by the agents since the exploration in A2C models is done through a random selection of actions given their probabilities calculated by the actor-network. The learning rate should be small enough so the agent can patiently visit different parts of the search space but not too small so that the learning process takes too long.

7-2-2-4 Pseudo-code of the training

The training procedure of A2C models is a straightforward procedure and can be found in the literature. Instead of explaining the common knowledge about the A2C models, we borrowed the pseudo-code for training the DMA2C models proposed for PL-IAM in Chapter 6.

7-2-3 Baseline for verification

The baseline used in this study is inspired by the NL-IAM system proposed for the Indiana, US bridges (Sinha *et al.*, 2009). This system uses a DTREE module for finding the right maintenance decision given expert judgments. Then, its RANK module prioritizes all proposed actions by the DTREE for all assets based on their costs and utilities. This prioritization is conducted with the IUC algorithm which is a deviation of the commonly used incremental benefit costs ratio. This algorithm selects projects with the highest utility/cost ratio within a given budget. The main difference between the implemented baseline in this study and the proposed system in the IBMS lies within the DTREE module. This study used the trained ML model proposed in Chapter 5 to derive the optimal life-cycle plans for each asset. Then, the same IAM system used that was used for the RL training was employed to find the expected agency costs, user costs, and utilities of the optimal strategies for each year. Finally, \$10M was considered the biannual budget constraint for the next 20 years. Then, an algorithm similar to the IUC was deployed to select the most appropriate intervention actions on the management horizon. The pseudo-code for this algorithm is provided in Algorithm 7.1.

Algorithm 7.1. Pseudo-code for deriving the baseline

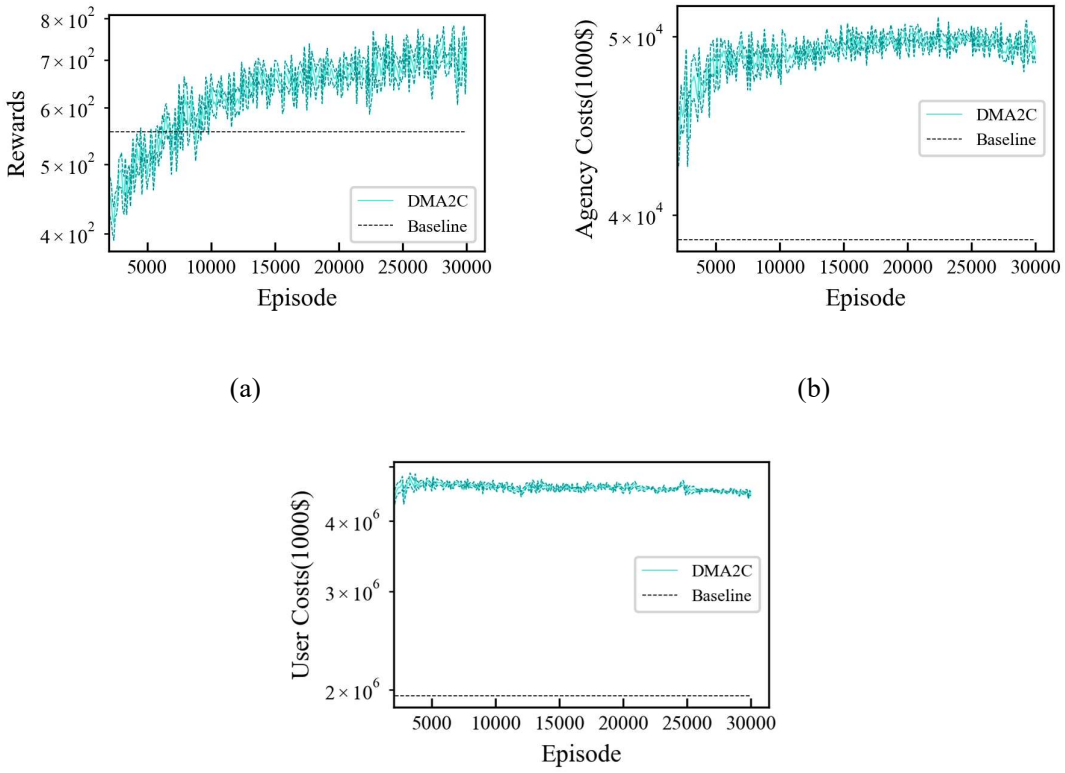
1:	Load models $\Theta_{O_1}: \Theta_{O_n}$	//Load models for predicting optimal actions for each element at each decision-making step
2:	Load models $\Theta_{L_1}: \Theta_{L_m}$	//Load models for predicting LCC and utilities

3:	**for each** asset i:	//Iterate over all assets
4:	find $\mathbf{M}_i^*$	//Find the optimal plans using Θ_O models
5:	**for** t **in** (1, T):	//Iterate over decision making steps in management horizon
6:	**find** C_{U_t}, C_{A_t}, U_t using $\Theta_L(\mathbf{M}^*)$	//Find user costs and agency costs for all assets at t
7:	**find** $R_{\mathbf{LCCA}_t}$	//Find the objective function value for each asset using optimal plans
8:	**Rank** assets based $\mathbf{R}_{\mathbf{LCCA}_t}$	//Rank assets based on their corresponding objective-value function
9:	$\mathbf{M}_j$= a set of actions to be done	//The improvement actions within the annual budget limitation of asset managers
10:	**Conduct** $\mathbf{M}_j$	//Conduct those actions within budget
11:	**Postpone** $\mathbf{M}_j$,	//Postpone the remaining action to the next decision-making step
12:	**find** $\bar{R}, \overline{C_A}, \overline{C_U}$	//Find the expected reward, agency costs, and user costs based on the finalized actions
13:	**return** $\bar{R}, \overline{C_A}, \overline{C_U}$	

7-3 Results and discussion

The characteristics of the computation machine used in this study are identical to that of the machine used in the studies for finding the baseline and the PL-IAM with RL. Although the computational machine is of less importance in scientific research, it can be important for comparing the computational time. This section presents the outcomes of the proposed framework by comparing the results with the baseline.

Figure 7.2, as one of the most basic analyses, depicts the learning convergence of the DMA2C models. These figures illustrate the increase in the expected rewards and decrease in the expected costs as the agent evolves by learning different strategies.



(a) (b)

(c)

Figure 7.2. Learning convergence of the DMA2C agents and expected (a) rewards, (b) agency costs, (c) user costs

Based on the results depicted in Figure 7.2, the DMA2C models could yield expected results (i.e., the stakeholders' utilities) of approximately 33% higher than the baseline. However, it should be noted that these higher expected results could be achieved by exploiting a bigger proportion of the budget which was not exploited by the baseline algorithm. Figure 7.2(c) demonstrates the significant relationship between user costs in comparison to the agency costs and rewards. The significance of this relationship lies within the approximately low influence of different strategies on the expected user costs for a network of large bridges with

relatively high ADT. This highlights the need for improving models for estimating road user costs, construction efficiency, and innovative measurements for reducing user costs.

Figure 7.4 depicts the likelihood of different actions that the DMA2C models take in 10,000 simulations. This figure can assist asset managers and decision-making bodies with managing resources in the long run based on the likelihood of actions for different assets. This figure, as a visualization tool for asset managers and decision-makers, depicts the likelihood of each action for each element each year. Since the decisions of the agents are a function of environmental and financial factors, the user and agency costs can vary widely. The histogram of the costs associated with the actions taken by the trained DMA2C models in 10,000 simulations is depicted in Figure 7.3.

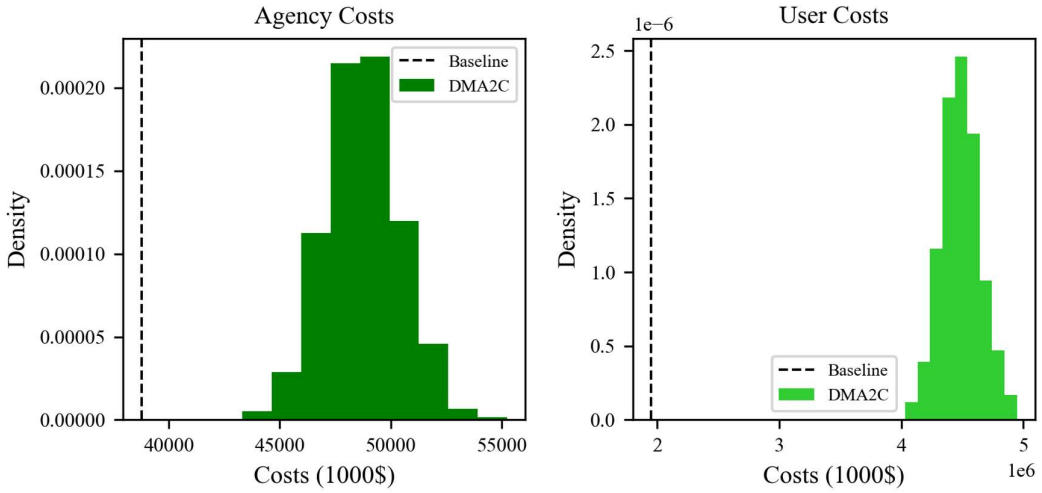


Figure 7.3. Histogram of agency and user costs of actions taken by DMA2C models

Figure 7.3 illustrates the expected exploitation of the DMA2C models from the available annual budgets. The important aspect of this figure lies within the costs distributions related to both strategies. While the expected costs of the baseline are a fixed amount using the costs models, there is approximately a 20% difference between the minimum and maximum of the costs given different occurrences in the life cycle of the network.

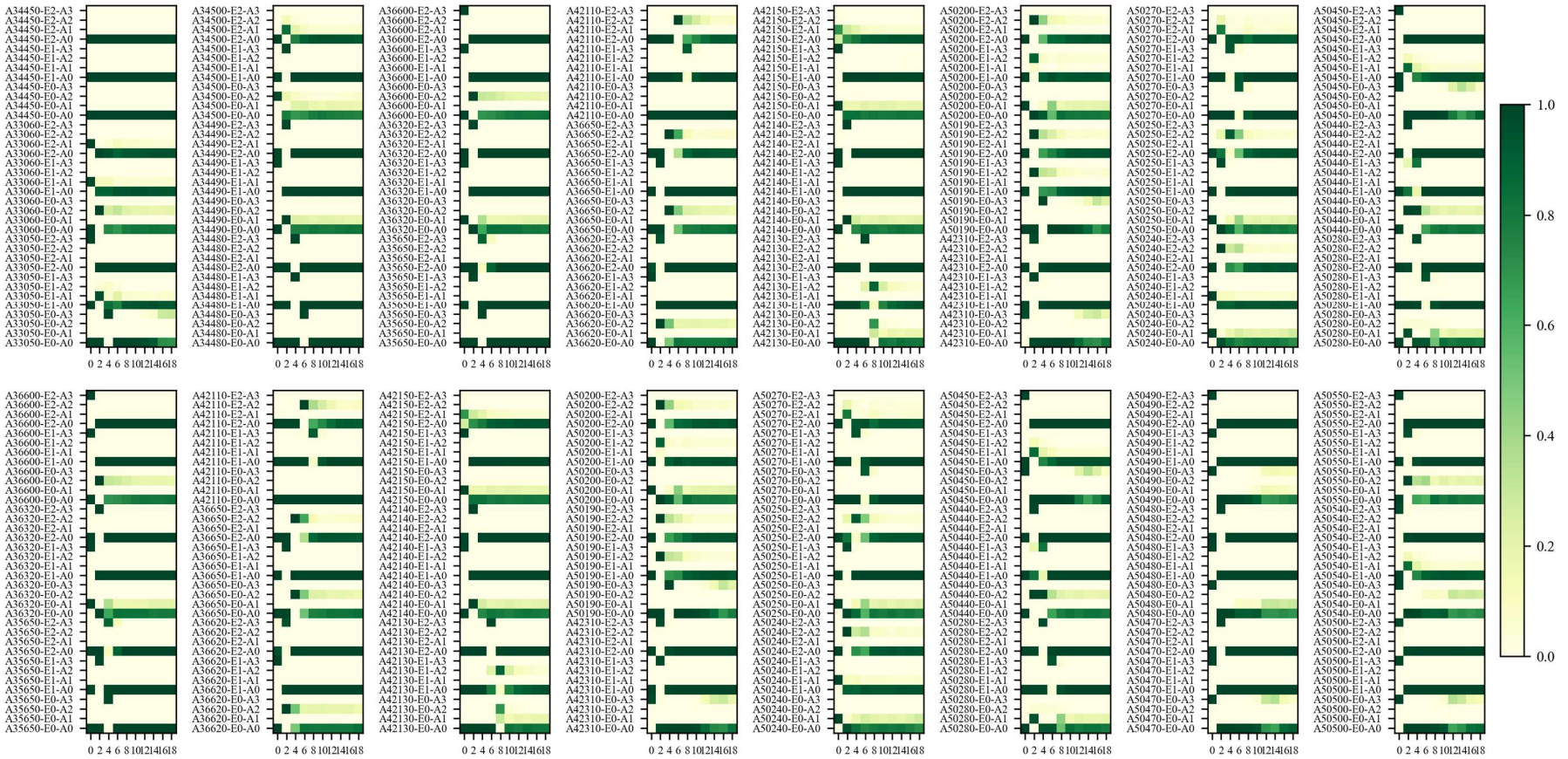


Figure 7.4. Likelihood of the intervention actions in the management horizon

The 20% difference in the agency costs means saving billions of dollars depending on the size of the network. A comparison of the costs associated with the baseline strategies and the DMA2C flexible plans is of less importance in this study. Fine-tuning the optimization settings and using more advanced baselines could add more value to these figures. However, these investigations lie beyond the scope of this study and are kept for future research.

Constructing the microworld and the DMA2C models, hyperparameter tuning, and training the models are time-consuming courses of action. In fact, each training session of the DMA2C models took about 50 hours. However, the results of the trained DMA2C models are potentially beneficial enough to ignore this computation time. In addition, this straightforward approach enables asset managers and researchers to incorporate various sources of uncertainties into NL-IAM and reach beneficial and flexible strategies. The proposed baseline approach, however, can be similarly used for NL-IAM while considering similar sources of uncertainties. The baseline and its strategies are derived at the expense of losing the flexibility in face of future uncertain phenomena like deterioration. The trade-off between the overhead computational cost and flexibility in decision making of RL-based (DMA2C) models and the proposed baseline should be made by asset managers, decision-making bodies, and researchers based on their needs and limitations.

Chapter 8 Conclusions and recommendations

8-1 Conclusions and contributions

This section provides a summary of the achievements, findings, and suggestions of this study. Aiming to properly allocate the limited funds to the maintenance of deteriorating assets, asset managers planning to rank a portfolio of assets or optimize life cycle MRR planning of an asset must use reliable and up-to-date tools. The available IAM systems could not be easily extended and used by asset managers, researchers, and decision-makers around the world. The most common framework for finding optimal life cycle improvement intervention actions (i.e., MCS and HOA framework) is far from applicability to a network of assets in real life due to its high computational costs. Moreover, the resulting optimal plans are subject to a key limitation: inflexibility to future uncertainties such as deterioration patterns, hazards, and fluctuating costs. The major objective of this dissertation was to propose a software platform and methodologies to tackle the abovementioned limitations. The following sub-sections focus on each objective presented in Chapter 1 of this dissertation.

8-1-1 Developing a general, extensible, and open-source infrastructure asset management software

All previously developed AMSs, heretofore, have been inextensible, costly, and written in less flexible programming languages. Motivated by the need to facilitate the research and practice in the asset management domain by both researchers and practitioners, this dissertation introduced the first freely accessible open-source platform for asset management which could be used as a stepping-stone in different asset management research and practical areas. The proposed theoretically well-founded computational platform, GIAMS, can serve both researchers and practitioners as a foundation for future research and practice in the asset management domain by facilitating the extension of current models and applying new models

in an easier manner. GIAMS is different from and superior to these asset management systems in some respects. First, GIAMS is open-source and extensible. This crucial characteristic of GIAMS can greatly accelerate progress in the field of asset management by enabling future researchers and practitioners to build upon each other's work and use the latest updates. Second, GIAMS is written in Python. Python language makes adopting advanced models related to asset management possible. To further illustrate, popular data science and ML platforms such as TensorFlow™ and Scikit-learn can be directly used in GIAMS. With the provided APIs for Python, practitioners can connect GIAMS to other scientific programming languages such as MATLAB for other advanced analyses. All in all, the popularity of Python among researchers and practitioners also paves the way for collaborating more efficiently. Third, GIAMS is freely accessible. Researchers and practitioners can clone the latest updates of the proposed GitHub repository. In addition, underprivileged communities and agencies that are incapable of affording licensed and expensive asset management systems could take advantage of this system. The extensibility and open-source nature of GIAMS promote the ability of researchers and practitioners to directly draw upon previous work when developing models for analyzing and optimizing the project-level life cycle of assets and network-level budget allocation for MRR actions. Either the implementation of advanced models or new optimization algorithms, more easily and rapidly conducted research in the asset management area is greatly enabled by the extensibility of the proposed platform. It is expected that through the open collaboration of researchers and practitioners over time, this software could be turned into an advanced and mature platform in the asset management research area.

In Chapter 3, the major building blocks of GIAMS (i.e., asset module, network module, life cycle analyzer module, and optimization module) were discussed. The current version of

GIAMS comprises different HOA(e.g., GA and IUC algorithm) for MRR optimization of assets by considering different sources of uncertainties such as degradation rate, fluctuations in costs over time, and consequences of hazards with uncertain magnitudes and frequencies. The utilization of GIAMS was illustrated through two examples based on real data of bridges in Indiana, US, and up-to-date realistic hazard and user costs models. All abovementioned sources of uncertainties were considered to optimize MRR planning of a bridge in a 20-year management horizon. Key concepts of open collaboration in software development are provided in Appendix A. Appendix B briefly discusses three examples for the development of GIAMS. In these examples, the simplicity of using GIAMS, testing a new idea, and contributing to the source code are shown. Moreover, extending GIAMS to a new area of asset management (i.e., prioritizing seismic retrofit of building structures (Talebiyan and Mahsuli, 2018)) and employing the concept of structural health monitoring in asset management (Straub and Faber, 2005; Orcesi and Frangopol, 2011; Thöns, 2018; Klerk *et al.*, 2019) are briefly discussed. Appendix C shows the directory tree and components of GIAMS, while Appendix D provides the information modeling of the illustrative examples as the current prototypes of GIAMS. All components of GIAMS are completely available through its repository (https://github.com/vd1371/GIAMS).

8-1-2 A methodology for learning the behavior of optimization algorithms by machine learning models and predicting the optimal maintenance planning of assets under uncertainties

Due to underlying sources of uncertainties such as occurrence time and magnitude of hazards, costs volatility in the future, and patterns of degradation of assets, PL-IAM studies have adopted sophisticated but time-consuming computational models to properly model different phenomena and optimize MRR actions in a management horizon. As a result, these models and asset management systems cannot be applied to each asset of a network because of their

high computational costs. Traditionally, asset management systems applied in real life usually use simplified models to assign an MRR plan to each asset in a network to derive the optimized maintenance interventions in a feasible amount of time. Intending to address a gap between literature and practice of PL-IAM stemming from the high computation time of advanced models, this dissertation put forward a new methodology to predict the LCCA results and optimized MRR plans given asset characteristics and MCS parameters instead of directly conducting optimization. Chapter 4 put forward a new methodology to reduce the LCCA computation time by estimating the LCCA results of an asset using DNN. DNN models were trained on datasets consisting of numerous synthesized bridges based on the US NBI with randomly generated MRR plans and corresponding LCCA results. Since three sources of stochastic uncertainties were present in the LCCA model, the LCCA results of each bridge were derived from the Monte Carlo simulation. The three DNN models for the user costs, agency costs, and utility had satisfactory prediction results (i.e., MAPE less than 2%, R-squared more than 0.98). DNN is an appealing option because it can: 1) be updated after observing new samples, 2) capture any degree of non-linearity in complex datasets, and 3) be trained on large datasets with reasonable computation time. Although this methodology has a relatively large overhead computational cost, the trained DNN models can yield similar results but hundreds of times faster than the MCS that is used in the MRR plan optimization of an asset by heuristic optimization algorithms. The proposed flexible methodology for estimating the LCCA results by training a DNN model provides the opportunity to use more complex models in the MRR optimization of each asset in a network. Filling the gap between academic and applied PL-IAM systems, this methodology enables practitioners and decision-makers to possibly identify more advantageous MRR strategies by incorporating probabilistic, non-linear, and other advanced techniques into their long-term planning.

8-1-3 A methodology for learning the behavior of optimization algorithms by machine learning models and predicting the optimal maintenance planning of assets under uncertainties

Next, Chapter 5 uses an augmented dataset consisting of thousands of data points each of which corresponds to a distinct asset, a set of MCS characteristics, and an optimal MRR plan. The MRR plans are derived by optimizing expected LCCA results calculated by MCS. The proposed methodology leverages the RF classification ML algorithm in asset management to predict MRR plans based on a semi-synthesized dataset consisting of more than 1.6 million bridges with their optimized MRR plans. RF model is generally an appealing choice of ML model given the imbalanced, multiclass, multioutput nature of this problem. The classification results of this methodology, with the accuracy of more than 95% and f1-score of more than 0.80 on the test set and with the accuracy of 89% and f1-score of 0.86 on highway bridges of Indiana with more than 4600 assets, demonstrate its practical applicability in such problems. Although this methodology required a relatively large overhead computational cost for the sampling optimization results, the hyperparameter tuning, and the ML training sessions, it can provide optimized MRR plans in a substantially shorter amount of time at the time of decision making. By filling the gap between the practice and literature on complex PL-IAM systems, the methodology presented in this dissertation provides the opportunity of obtaining possibly more beneficial and comprehensive MRR plans for all assets in a network without sacrificing detailed models. The proposed methodology enables practitioners and researchers to upscale complex PL-IAM systems, which consider multiple sources of uncertainties with probabilistic and non-linear models, to be applied to all assets in a network.

8-1-4 Developing a holistic framework for adopting reinforcement learning methodologies and utilizing the long-established aspects of infrastructure asset management under uncertainty in reinforcement learning-based studies

Ideally, IAM systems should be able to capture underlying uncertainties such as stochastic deterioration patterns, unprecedented earthquakes, and fluctuating costs and inform decision-makers about long-term optimal plans. Recent studies have defined IAM projects as MDP problems to add flexibility to their decision-making in face of uncertainties such as deterioration patterns. Chapter 6 sought to address a gap in the existing body of IAM knowledge emanated from (1) the need to add more than one flexibility factor based on multiple uncertainties, and (2) discrepancies between the proposed innovative theoretical-oriented RL-based studies and IAM practice. The proposed theoretically well-founded yet practice-oriented methodology contributes to the existing body of IAM knowledge by 1) Adding multiple flexibility factors in terms of CR of assets, market status, and projected costs while incorporating several sources of uncertainties (e.g., unprecedented hazards and their consequences, fluctuations in costs, and uncertain deterioration patterns), 2) taking steps forward toward providing a more practical RL-based IAM framework by utilizing managerial considerations such as stakeholders' utilities, budget constraints, intervention actions preferences, and 3) providing a holistic framework for future RL-based IAM studies and practice from early development and training to validation. A high-level introduction to RL, some training techniques, and MARL has been provided in this study. Then, we provided further details about how to develop IAM systems, and then, upgrade them to work as a microworld for interacting with RL models. Next, training aspects of MARL models with respect to the context of IAM are discussed. Finally, and most importantly, validation of MARL results is fully discussed. Accordingly, we developed a microworld on top of a

previously developed and freely available BMS and took two common approaches (DQN and A2C) toward training different agents each of which is responsible for taking preventive actions for elements (i.e., deck, superstructure, substructure) of a real bridge in Indiana, US. Finally, the flexible strategies of the trained MARL models were compared with a rigid optimal plan based on the results of a previous study. The results show that the trained A2C agent can potentially increase the stakeholders' utilities by 14% while reducing the agency costs and user costs by almost 6%. Taking a different policy through the training process, DQN agents tried to keep maintaining the bridge in the management horizon leading to a 35% reduction in agency costs and a 31% increase in user costs. Finally, the RL models were trained in a shorter amount of time highlighting their computational merits. The provided flexibility by defining the IAM problem as an MDP and training MARL for taking optimal intervention strategies can assist researchers and practitioners in taking closer to real decisions. Notably, the trained MARL models can take decisions in a far shorter time in comparison to other complex MCS-HOA-based IAM frameworks that have been proposed to consider similar sources of uncertainties.

8-1-5 Network-level infrastructure asset management under complex uncertainty using reinforcement learning methods and developing recommended solutions for the curse of dimensionality and reward engineering

Motivated by the need for an RL-based NL-IAM framework and in line with a relatively recent line of research, Chapter 7 of this study aims at proposing one of the first NL-IAM RL-based frameworks to solve a large NL-IAM problem being formulated by MDPs. Keeping the practical aspects, such as computational time at the decision-making stages, into consideration, Chapter 7 proposed a new generation of frameworks for NL-IAM as a stepping stone for future studies in this domain. The inherent characteristics of the proposed

framework make it advantageous to its counterparts by 1) incorporating the effects and consequences of multiple sources of uncertainties such as deterioration, earthquake, and costs into the decision making process, 2) informing decision-makers and asset managers of potentially more beneficial strategies, in comparison to the conventional algorithms, given the updated information about uncertainties, 3) taking optimal decisions in a negligible amount of time once the agents are trained. Apart from the contributions of the proposed NL-IAM framework, this study can potentially contribute to general multi-agent large-scale RL-based studies by proposing rudimentary, yet effective, solutions for reducing the dire consequences of the curse of dimensionality. Last, but not least, the proposed baseline for validating the results of the DMA2C models can be separately used for NL-IAM applications.

Chapter 7 started by providing an intuitive and high-level introduction to the concept of RL, A2C models, and MARL approaches. Then, the methodology section discusses the whole process of developing microworlds and training DMA2C models. This section highlights two curses of large MDPs problems (i.e., the curse of dimensionality) and provided solutions for these problems. The final part of this section was allocated to the validation approach for the results of the trained DMA2C agents. The applicability and performance of the proposed framework in this study were demonstrated by a subset of the bridge in Indiana, US bridges. After discussing the employed BMS, details of the DMA2C training (e.g., feature engineering, reward engineering, hyperparameters, and the training procedure), and the baseline, a discussion was provided on the derived results. The results of this study showed that it is possible to increase stakeholders' utilities by 33%, albeit at the expense of a more costly strategy, in comparison to the presented baseline with identical constraints and optimization settings. The significance of the results mainly lies within the distribution of the costs of the trained DMA2C models rather than their relation to the fixed costs of the

baseline. In detail, the minimum expected costs are 20% less than the maximum expected costs given the different uncertainties in the life cycle of the network. More advanced baselines or other optimization settings could minimize the difference between the results of the baseline and the DMA2C models. The trained agents are readily available to take intervention decisions and to be used in practice. Although training the agents and constructing this framework was a relatively time-consuming process, the decisions are taken by the agents almost instantly.

8-2 Limitations and Future Research

The proposed methodologies and the developed platform are in their primitive stages and are subject to some shared limitations. Although the applicability of the frameworks the GIAMS was showcased using a BMS case study, there is no known theoretical barrier against extending their application into other similar IAM systems such as pavement management and railway management Future research is required to implement models describing other phenomena governing different assets such as pavement and sewage systems that could be developed and used in GIAMS. In other words, the proposed platform can also be further developed and extended to be applied to other types of assets that need to be managed by agencies, municipalities, and other decision-makers. Also, future research could evaluate the applicability of the proposed methodologies for reducing computation time and adding flexibility to the IAM in other fields of asset management such as pavement management systems.

Another important limitation of GIAMS and the proposed methodologies is the modeling approach toward uncertain phenomena such as earthquake occurrence, user costs volatility, and elements' deterioration for representing real-world phenomena. The uncertainty models used for modeling hazards, costs, and deterioration could be enhanced to accurately

characterize these stochastic processes. There is also a need for further investigation on including other social, environmental, socioeconomic, and in general sustainability measures in decision-making. As a result, there is abundant room for improvement of this methodology in terms of the complexity of models and the validity of results. However, without the loss of generality of the proposed methodology, more complex and advanced models could be used to imitate the underlying phenomena in IAM for various asset types.

Since the major parts of the collected data in this study are firsthand and collected from reliable sources and the implemented models are adopted from previously published studies in highly reputable journals, they were not validated in this study. Although the main purpose of this study is to push forward the boundaries of theoretical IAM systems, there is a need for further validation of the collected results and models before real-life application. Therefore, validation of data and models are strongly recommended before using the proposed framework and applying the proposed methodologies.

The developed platform in Chapter 3, GIAMS, is currently in its embryonic stage. Apart from the shared limitation among the five analytical chapters of this study, investigating and developing other optimization algorithms (e.g., Lagrangian relaxation) in GIAMS is another possible area of future research would be to. Also, there is abundant room for further research in the implementation of advanced models (e.g., ML models for predicting deterioration) and developing them in GIAMS.

Theoretically, more data samples could improve DNN and RF prediction results in Chapter 4 and Chapter 5. Therefore, synthesizing more assets could lead to constructing near-perfect ML models. The trade-off between the computation time of training these models and MRR optimization of all assets will be more appealing if the overhead computation time of training ML models could be reduced. This study is an early attempt at upscaling complex

PL-IAM systems. Future studies can focus on improving the overhead computation time and accuracy of this methodology by applying other advanced computer science and ML techniques. That said, DNN and RF models are commonly known as black boxes that cannot provide further knowledge about the data and system. There is a need for the implementation of model explainers or explainable models for a better understanding of the role of input variables in the models. Also, there is a need for the integration of simulation results with the ML prediction models in the pre-studied scenarios where ML models would lead to less accurate results.

The proposed frameworks in Chapter 6 and Chapter 7 are steppingstones and could be further enhanced in a number of respects. Perfect tuning of the hyperparameters was not in the scope of this study. Future studies could focus on providing guidelines toward reward engineering and hyperparameter tuning of both models and RL training. Lastly, Chapter 6 covered project-level decision-making and Chapter 7 focused on 48 assets given the limitations of the available computational machines. There is a need for detailed and in-depth analyses for extending such applications to mega networks with thousands of assets and addressing their corresponding challenges. The proposed approaches for tackling the curse of dimensionality are rather rudimentary. Further investigation and more effective approaches are required to limit the adverse effects of the curse of dimensionality. Finally, the RL approach could be updated by more advanced techniques and models to speed up and optimize the learning efficiency of agents.

Future research on the mentioned limitations and broader opportunities are therefore recommended.

Appendices

Appendix A

This section defines basic terminology and concepts for open collaboration in further developing the platform by volunteer researchers. Open collaboration in software development has been enabled by using distributed version-control systems that facilitate software development among developers. Git is one of the most popular version control systems and through the help of online and free access repositories (e.g., GitHub (*Github*, 2020)), which are usually called remote repositories, forms the foundation of numerous projects. Collaborators will need to have the git installed on their systems and own a personal GitHub page. Four common steps executed in git and GitHub, namely fork, clone, edit, and pull request, involve contributing and developing with an open collaboration mindset. These steps will be summarized in the following paragraphs:

Step 1: Fork

Forking refers to the process in which one collaborator makes a copy of others' repositories (e.g., GIAMS repository) to her Github page. This step can be easily carried out using features of GitHub available on the repository page.

Step 2: Clone

Cloning refers to the process of downloading the source code to a folder on the local machine for further development. This is usually performed as the second step in contributing to others' work.

Step 3: Edit

The editing process involves developing code, committing the code to the version control system (git), and pushing the updated code to the remote repository. In short, saving the changes in the git is called committing, and uploading the code to the remote repository is

called pushing the code. Git enables developers to change and update files in branches instead of directly changing the code. The main branch, which is called the master branch in git, usually remains untouched during the development. Other user-defined branches will be assigned for further development of the framework. These branches will finally be merged into the master branch by the principal contributors. Figure A. 1 depicts the notion of branches in software development in an abstract form.

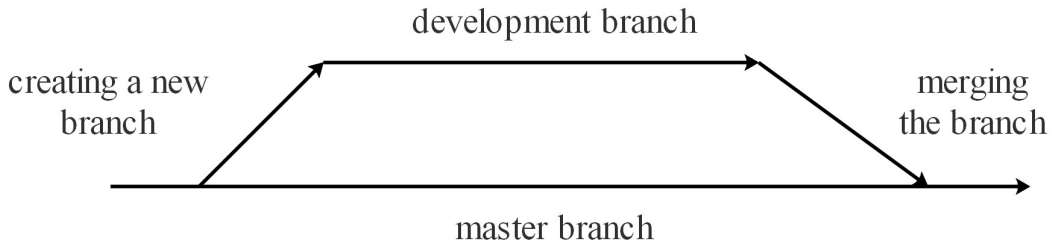


Figure A. 1. Git branches in abstract form

Step 4: Pull request

After editing and pushing the changes, the contributor should inform the main contributor to the project about the updates on a project. The submission of a pull request refers to the process of asking the repository owner to merge the updated code with the original source code. This task is initiated on the contributor's own GitHub page after pushing the code to the remote repository. The main developers or other contributors will then review the updates, check every single added unit, and validate the newly developed code. In case of validity, the new updates will be merged with GIAMS.

Apart from directly contributing to the source code, researchers could raise issues or make feature requests in the *Issues* section of the Github repositories. The conventional response of the contributors of a repository is replying to communications regarding issues or feature requests and updating the code.

Appendix B

This section briefly discusses three examples that are other extensions for GIAMS. These examples could be used as simple guides for future developments and extensions.

Example1: A new deterioration model

This example illustrates the process of implementing a new deterioration model and contributing to the development of GIAMS. In this example, an imaginary researcher called "GIAMS Researcher" has developed a deterioration model that can predict the future CR of bridge elements located in Indiana as a function of bridges' latitude (Lat.) and longitude (Long.) as well as previous condition of elements. Table B. 1 describes all the required steps for contributing her model, referred to here as *LatLongFunc,* to the GIAMS repository to be accessed and used publicly.

Table B. 1. The steps for adding a new deterioration model to GIAMS

	Task	Code
1	Forking the GIAMS repository to her page	
2	Cloning the GIAMS in a folder	In git bash: *git clone https://github.com/giams_researchers/GIAMS.git*
3.1	Adding the Lat. And Long. of bridges to Indiana .CSV file to columns 21 and 22	In Network/Networks/INDIANA2019.csv
3.2	Adding a method to set these attributes to the Bridge	In Asset/AssetTypes/Bridge.py: def set_latitude_longitude(self, lat, long): self.lat, self.long = lat, long
3.3	Adding the deterioration model	In Asset/Elements/Deterioration: - Copying and renaming one of the previously provided models to

		LatLong.py - Changing the name of the class to *LatLong* - Changing the *predict_condition* of the *LatLong.py:* def predict_condition(previous_condition): return **LatLongFunc**(self.asset.lat, self.asset.long, previous_condition) In the *load_asset* method of Network/IndianaNetwork.py: lat, long = asset_info[21:23] asset.set_latitude_longitude(lat, long) *element.set_deterioration_model(LatLong())
3.4	Updating the network loader	
3.5	Pushing the changes to the remote repository	In git bash: - git branch langlat_branch - git add . - git commit git push origin langlat_branch
4	Making a pull request on her repository branch	

The main contributors will then evaluate the pull request and in case of validity will merge it with the source code. In this example, the "GIAMS Researcher" could use the readily developed platform in several simple steps without writing another asset management system from scratch. This would save enormous time and effort expended by asset management researchers and practitioners. Although not all developments and modifications are as simple as this example, the general procedure remains intact and the proposed framework remains applicable.

Example2: Prioritizing seismic retrofit of building structures

Taking structural seismic retrofit planning of a portfolio of buildings in a state/province as an example, this section discusses the extent of required modifications for new area applications to be developed in GIAMS. The modification of GIAMS modules varies depending on the nature of the problem. Analyzing new problems would require more rigorous modifications in comparison to example 1. A number of new modules should be developed for this purpose while the rest of GIAMS' modules could be directly or with minor modifications to be used. For example, a new module for buildings should be developed, while some other modules like hazard generators could be readily used with minor or no modifications. Table B. 2 summarizes the extent of modification and development of modules that are required for this example.

Table B. 2. The extent of changes required for GIAMS' modules for structural seismic retrofit

New modules required		**Minor modifications/No Change**	
Asset/AssetTypes	/Building	Asset/Elements	/Condition rating
			/Deterioration models
Asset/Elements	/Structure	Asset/HazardModels	/Generator
	/AgencyCost		/Recovery
	/Utility		
Asset/HazardModels	/Loss	Asset/MRRModels	/MRRTwoActions
	/Response		/MRREffectiveness
Asset/UserCostModels	/UserCosts	LifeCycleAnalyzer	/Simulator
			/LCA
Network	/BuildingPortfolio	Optimizer	/GA
			/IUC
LifeCycleAnalyzer/Simulator	/RiskAnalyzer		
Optimizer	/Rank		

The developed dummy models for the abovementioned problem could be readily found in the GIAMS repository. During the first time extension of GIAMS for retrofit prioritization of

buildings, several new modules should be developed for this purpose, though the majority of GIAMS' modules could be used directly or with minor modifications. After the first-time implementation of retrofit prioritization of buildings, the extension of GIAMS for similar purposes would be merely limited to minor modifications of its modules. As a result, the efforts of researchers, practitioners, and collaborators that will be built upon each other will make future modifications and extensions of GIAMS easier.

Example3: Maintenance optimization by considering structural health monitoring

Maintenance optimization of assets based on structural health monitoring (SHM) results is another common task in asset management. Similar to example 2, this section provides a brief guideline for adapting GIAMS to consider SHM updates in the life cycle of assets with a focus on bridge structures. SHM results usually affect the previously designated MRR decisions, deterioration models, and future inspection planning (Straub and Faber, 2005; Orcesi and Frangopol, 2011; Thöns, 2018; Klerk *et al.*, 2019). Therefore, a few new modules such as a new simulator should be written from scratch, a few modules should be modified, while the rest of the framework could be directly used. Table B. 3 provides a summary of the extent of GIAMS' modifications that are required for this example.

Table B. 3. The required modifications for GIAMS' modules for adopting SHM

New modules required		**Minor modifications/No Change**	
Asset/MRRModels	/SHMActions	Asset/Elements	/Deterioration models
	/SHMEffectiveness		
Asset/Elements	/SHMCost	Asset/UserCostModels	/UserCosts
LifeCycleAnalyzer	/SHMSimulator	Asset/HazardModels	/Generator
			/Recovery
			/Loss
			/Response
Network	/SHMNetwork	LifeCycleAnalyzer	/LCA

	Optimizer	/GA
		/IUC

Dummy models are used in the abovementioned modules, which could be found in the GIAMS repository, to illustrate the extensibility of GIAMS. Similar to example 2, after the first-time development of structural health monitoring in GIAMS, the extension of GIAMS for more advanced implementations of structural health monitoring will be limited to minor modifications. Although the current version of GIAMS is not designed to cover all domains of IAM, its design pattern and accessibility will make further developments simpler than developing a new IAM system from scratch.

Appendix C

The directory tree of the GIAMS repository is summarized as follows:

```
GIAMS
|---Asset/
|       |---AssetTypes/
|               |---Bridge
|               |---other assets…
|       |---Elements/
|               |---AgencyCost/
|                       |--- BaseAgencyCost
|                       |--- other models…
|               |---ConditionRating/
|                       |--- BaseConditionRating
|                       |--- other models…
|               |---Deterioration/
|                       |--- BaseDeterioration
|                       |--- other models…
|               |---Utility/
|                       |--- BaseUtility
|                       |--- other models…
|               |---BaseElement
|               |---BridgeElement
|       |---HazardModels/
|               |---Generator/
|                       |--- BaseHazard
|                       |--- other models…
|               |---Loss/
|                       |--- BaseHazardLoss
|                       |--- other models…
|               |---Recovery/
|                       |--- BaseRecovery
|                       |--- other models…
|               |---Response/
|                       |--- BaseResponse
|                       |--- other models…
|               |---HazardModel
|       |---MRRModels/
|               |---EffectivenessModels/
|                       |--- BaseMRREffectiveness
|                       |--- other models…
|               |---BaseMRRPlan
|               |---other models
|       |---UserCostModels/
|               |--- BaseUserCost
|               |--- other models…
.
.
.
.
.
|---LifeCycleAnalyzer/
|       |---Simulators/
|               |---BaseSimulator
|               |---other models…
|       |---BaseLCA
|       |---LCA
|       |---other models…
|---Network/
|       |---Networks/
|               |---networks csv files
|       |---BaseNetwork
|       |---other models…
|---Optimizer/
|       |---GA
|       |---IUC
|---reports/
|---utils/
```

Figure C. 1. Directory tree of GIAMS

Appendix D

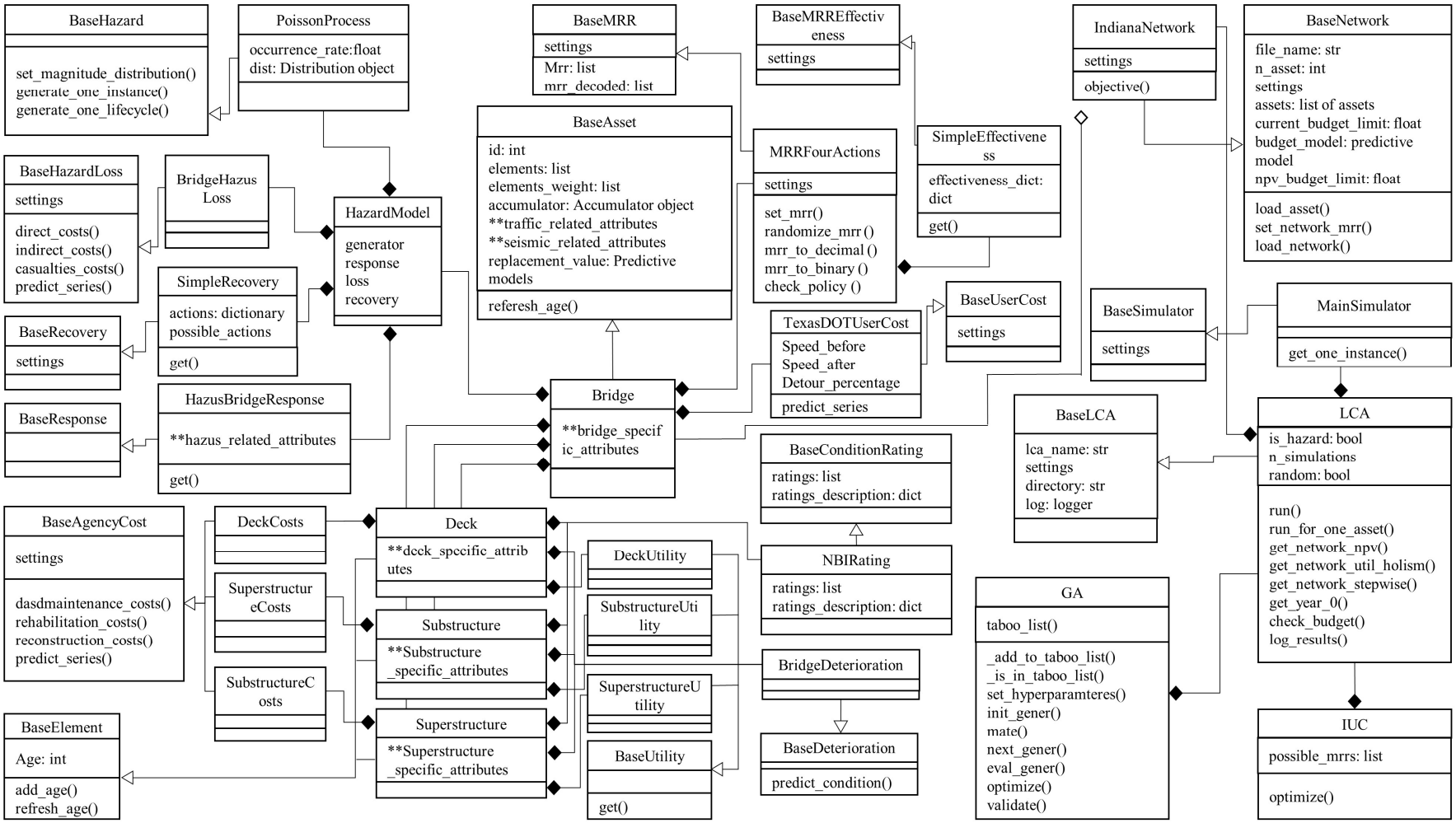


Figure D. 1. Information modeling of the illustrative examples

Appendix E

Algorithm E1. Algorithm of the LCCA module in the case study

1:	**Input:** Range of bridge characteristics and simulation parameters	// Table 4.3
2:	$C_{U_j}, C_{A_j}, U_j = 0, 0, 0$	// C_U, C_A, U_j: holders for user costs, agency costs, and utilities for each element j
3:	**M** = A synthesized MRR plan	// Synthesizing an MRR plan
4:	$\mathbf{A_p}$ = A synthesized bridge	// Using values of Table 4.3
5:	$\mathbf{S_{MCS}}$ = Synthesized simulation parameters	// Using values of Table 4.3
6:	**for** $n \in \{0, 1, \dots, N\}$ **do**:	// N: Number of simulations
7:	$C'_{U_j}, C'_{U_j}, U'_j = 0, 0, 0$	// C'_{U_j}, C'_{A_j}, U'_j: holders for user costs, agency costs, and utilities for each element j in each round of simulation
8:	**H** = A generated hazard sample	// Based on hazard distribution characteristics of $\mathbf{S_{MCS}}$ parameters and Eq. (3.1)
9:	**for** t $\in \{0, 2, \dots, T\}$ **do**:	// T: Management horizon in $\mathbf{S_{MCS}}$ parameters
10:	**for** $j \in \{elements\}$:	// for each element in the bridge
11:	**if** t **in H**:	
12:	**find** S'_j	// S': State of the bridge in response to hazards given $\mathbf{A_p}$ parameters and Eq. (4.5)
13:	**RC** = **find** recovery plans	// Finding proper recovery actions based on the recovery model
14:	**if** t **in RC**:	
15:	**find** C'_{A_j}, C'_{U_j}	// Find costs based on Eq. (4.6)– Eq. (4.10) and corresponding models[1, 2] given $\mathbf{S_{MCS}}$ and $\mathbf{A_p}$ parameters

16:	**find** S'_j	// S': State of the bridge element after recovery based on the recovery model and guidelines in literature[3]
17:	**elif** t **in M** and action != DONOT:	
18:	**find** C'_{A_j}, C'_{U_j}, U'_j	// Find costs based on Eq. (4.6)– Eq. (4.10) and corresponding models[1, 2] and find utilities based on Eq. (4.11) to Eq. (4.14) and corresponding models[4] given $\mathbf{S_{MCS}}$ and $\mathbf{A_p}$ parameters
19:	**find** S'_j	// S': State of the bridge element after MRR actions based on the recovery model and guidelines in literature[3]
20:	**else:**	
21:	**find** S'_j	// S': State after degradation based on degradation model[1] and $\mathbf{A_p}$
22:	$C'_{U_j} += C'_{U_j}/(1+r)^t$ $C'_{A_j} += C'_{A_j}/(1+r)^t$ $U'_j += U'_j/(1+r)^t$	// Adding discounted user costs, agency costs, and utility of actions for each element to its corresponding holder in each round simulation
23:	$S_j = S'_j$	// Update state of bridge element
24:	age$_j$ = **Update** age of element j	// Update age of each element given the type of actions
25:	**end for**	
26:	$C_{U_j} = (C_{U_j} \times (n-1) + C'_{U_j})/n$ $C_{A_j} = (C_{A_j} \times (n-1) + C'_{A_j})/n$ $U_j = (U_j \times (n-1) + U'_j)/n$	// Updating user costs, agency costs, and utilities for finding the average results of MCS
27:	**end for**	
28:	**return** C_{U_j}, C_{A_j}, U_j	

Note: 1- (Sinha *et al.*, 2009), 2- (TexasDOT, 2020), 3-(Hawk and Small, 1998)**,** 4- (Bai *et al.*, 2013)

Appendix F

Figure F.1 illustrates the details of the implemented PL-IAM system. Delving through all these details and discussing every single part of them require another full-length study and lie beyond the scope of this paper. Further details and information on this asset management system can be found in (Asghari and Hsu, 2021).

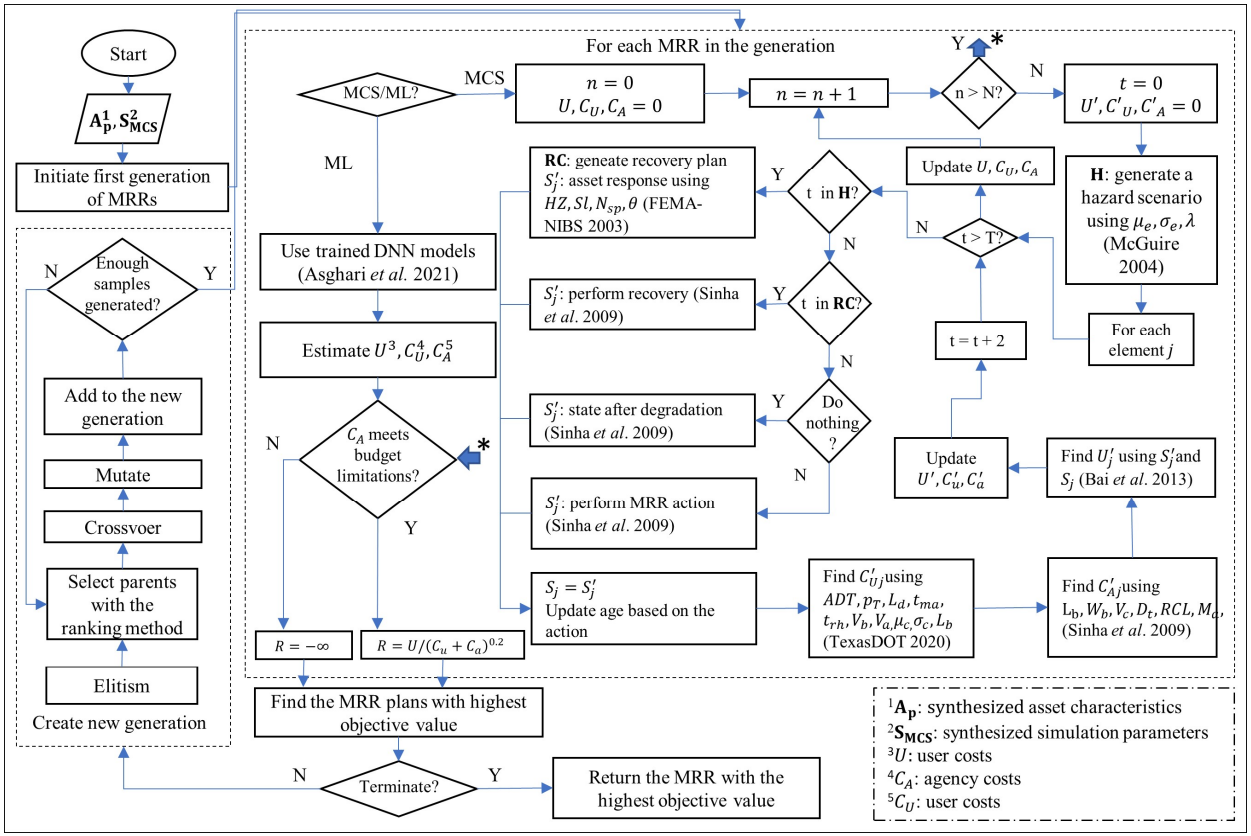


Figure F. 1. Implemented PL-IAM models and calculations in detail

References

AASHTO (2015) *Manual for bridge element inspection*. American association of state highway and transportation officials.

Abadi, M. *et al.* (2016) 'TensorFlow: Large-scale machine learning on heterogeneous systems', *arXiv preprint arXiv:1603.04467*. doi: 10.1016/0076-6879(83)01039-3.

Adam, B. and Smith, I. (2008) 'Active tensegrity: A control framework for an adaptive civil-engineering structure', *Computers & Structures*, 86, pp. 2215–2223. doi: 10.1016/j.compstruc.2008.05.006.

Ahmad, A., Amr, K. and Mario, V. (2018) 'New multiobjective optimization approach to rehabilitate and maintain sewer networks based on whole lifecycle behavior', *Journal of Computing in Civil Engineering*. American Society of Civil Engineers, 32(1), p. 4017069. doi: 10.1061/(ASCE)CP.1943-5487.0000715.

Ahmad, A., Mario, V. and Amr, K. (2018) 'New approach for critical pipe prioritization in wastewater asset management planning', *Journal of Computing in Civil Engineering*. American Society of Civil Engineers, 32(5), p. 4018044. doi: 10.1061/(ASCE)CP.1943-5487.0000784.

Akanmu, A. A. *et al.* (2020) 'Cyber-physical postural training system for construction workers', *Automation in Construction*. Elsevier B.V., 117, p. 103272. doi: 10.1016/j.autcon.2020.103272.

Alireza, M., Luis, A.-J. and Fuzhan, N. (2020) 'Reliable, effective, and sustainable urban railways: A model for optimal planning and asset management', *Journal of Construction Engineering and Management*. American Society of Civil Engineers, 146(6), p. 4020057. doi: 10.1061/(ASCE)CO.1943-7862.0001839.

Alysson, M., M., F. D. and Liang, L. (2018) 'Bridge adaptation and management under

climate change uncertainties: A review', *Natural Hazards Review*. American Society of Civil Engineers, 19(1), p. 4017023. doi: 10.1061/(ASCE)NH.1527-6996.0000270.

American Society of Civil Engineers (2021) *ASCE's 2021 infrastructure report card*. Available at: https://infrastructurereportcard.org/wp-content/uploads/2020/12/National_IRC_2021-report.pdf.

An, Y. *et al.* (2021) 'A reinforcement learning approach for control of window behavior to reduce indoor PM2.5 concentrations in naturally ventilated buildings', *Building and Environment*, 200, p. 107978. doi: https://doi.org/10.1016/j.buildenv.2021.107978.

Andriotis, C. P. and Papakonstantinou, K. G. (2019) 'Managing engineering systems with large state and action spaces through deep reinforcement learning', *Reliability Engineering & System Safety*, 191, p. 106483. doi: https://doi.org/10.1016/j.ress.2019.04.036.

Andriotis, C. P. and Papakonstantinou, K. G. (2021) 'Deep reinforcement learning driven inspection and maintenance planning under incomplete information and constraints', *Reliability Engineering & System Safety*, 212, p. 107551. doi: https://doi.org/10.1016/j.ress.2021.107551.

Apolinarska, A. A. *et al.* (2021) 'Robotic assembly of timber joints using reinforcement learning', *Automation in Construction*. Elsevier B.V., 125, p. 103569. doi: 10.1016/j.autcon.2021.103569.

Arif, F., Bayraktar, M. E. and Chowdhury, A. G. (2015) 'Decision support framework for infrastructure maintenance investment decision making', *Journal of Management in Engineering*, 32(1), p. 04015030. doi: 10.1061/(asce)me.1943-5479.0000372.

Asghari, V. (2020) *XProject*. Available at: https://github.com/vd1371/XProject (Accessed: 2 June 2021).

Asghari, V. (2021) *EstOpt*. Available at: https://github.com/vd1371/EstOpt (Accessed: 28

October 2021).

Asghari, V. and Hsu, S.-C. (2020) *GIAMS - A General Infrastructure Asset Management System*. Available at: https://github.com/vd1371/GIAMS (Accessed: 6 January 2021).

Asghari, V. and Hsu, S.-C. (2021) 'An open-source and extensible platform for general infrastructure asset management system', *Automation in Construction*. Elsevier B.V., 127(April), p. 103692. doi: 10.1016/j.autcon.2021.103692.

Asghari, V., Hsu, S.-C. and Wei, H.-H. (2021) 'Expediting life cycle cost analysis of infrastructure assets under multiple uncertainties by deep neural networks', *Journal of Management in Engineering*. doi: 10.1061/(ASCE)ME.1943-5479.0000950.

Asghari, V., Kashani, H. and Hsu, S.-C. (2021) 'Optimal timing of the seismic vulnerability reduction measures using real options analysis', *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*. doi: 10.1061/AJRUA6.0001146.

Asghari, V., Leung, Y. F. and Hsu, S.-C. (2020) 'Deep neural network based framework for complex correlations in engineering metrics', *Advanced Engineering Informatics*. Elsevier, 44(February), p. 101058. doi: 10.1016/j.aei.2020.101058.

Ashkan, S. M. *et al.* (2020) 'Estimating stripping of asphalt coating using k-Means clustering and machine learning–based classification', *Journal of Computing in Civil Engineering*. American Society of Civil Engineers, 34(1), p. 4019044. doi: 10.1061/(ASCE)CP.1943-5487.0000864.

Ashuri, B. *et al.* (2012) 'Risk-neutral pricing approach for evaluating BOT highway projects with government minimum revenue guarantee options', *Journal of Construction Engineering and Management*, 138(4), pp. 545–557. doi: 10.1061/(asce)co.1943-7862.0000447.

Aven, T. and Ford, D. N. (2004) 'Valuation of a well construction project with a flexible

project plan, comparison of real option pricing and Monte Carlo simulation'.

Bai, Q. *et al.* (2013) 'Multiobjective optimization for project selection in network-level bridge management incorporating decision-maker's preference using the concept of holism', *Journal of Bridge Engineering*, 18(9), pp. 879–889. doi: 10.1061/(ASCE)BE.1943-5592.0000428.

Biondini, F. and Frangopol, D. M. (2016) 'Life-cycle performance of deteriorating structural systems under uncertainty: Review', *Journal of Structural Engineering*, 142(9), p. F4016001. doi: 10.1061/(ASCE)ST.1943-541X.0001544.

Bowes, B. D. *et al.* (2021) 'Flood mitigation in coastal urban catchments using real-time stormwater infrastructure control and reinforcement learning', *Journal of Hydroinformatics*, 23(3), pp. 529–547. doi: 10.2166/HYDRO.2020.080.

Branco, P., Torgo, L. and Ribeiro, R. P. (2019) 'Pre-processing approaches for imbalanced distributions in regression', *Neurocomputing*, 343, pp. 76–99. doi: https://doi.org/10.1016/j.neucom.2018.11.100.

Brandi, S. *et al.* (2020) 'Deep reinforcement learning to optimise indoor temperature control and heating energy consumption in buildings', *Energy and Buildings*. Elsevier Ltd, 224, p. 110225. doi: 10.1016/j.enbuild.2020.110225.

Breiman, L. (2001) 'Random forests', *Machine Learning*, 45(1), pp. 5–32. doi: 10.1017/CBO9781107415324.004.

Breiman, L. *et al.* (2017) *Classification and regression trees*. Boca Raton: Chapman & Hall/CRC.

Brennan, M. J. and Schwartz, E. S. (1976) 'The pricing of equity-linked life insurance policies with an asset value guarantee', *Journal of Financial Economics*, 3(3), pp. 195–213. doi: https://doi.org/10.1016/0304-405X(76)90003-9.

Brockman, G. *et al.* (2016) 'OpenAI Gym'. doi: arXiv:1606.01540.

Campos, R. M. *et al.* (2021) 'Operational wave Forecast selection in the Atlantic ocean using random forests', *Journal of Marine Science and Engineering*. doi: 10.3390/jmse9030298.

Capasso, G., Gianfrate, G. and Spinelli, M. (2020) 'Climate change and credit risk', *Journal of Cleaner Production*, 266, p. 121634. doi: https://doi.org/10.1016/j.jclepro.2020.121634.

Cardoso, M. A., Almeida, M. C. and Santos Silva, M. (2016) 'Sewer asset management planning – implementation of a structured approach in wastewater utilities', *Urban Water Journal*, 13(1), pp. 15–27. doi: 10.1080/1573062X.2015.1076859.

Chawla, N. *et al.* (2002) 'SMOTE: Synthetic minority over-sampling technique', *Journal of artificial intelligence research*, 16, pp. 321–357. doi: 10.1613/jair.953.

Chembe, C. *et al.* (2019) 'Infrastructure based spectrum sensing scheme in VANET using reinforcement learning', *Vehicular Communications*. Elsevier Inc., 18, p. 100161. doi: 10.1016/j.vehcom.2019.100161.

Chemingui, Y., Gastli, A. and Ellabban, O. (2020) 'Reinforcement learning-based school energy management system', *Energies*, 13, p. 6354. doi: 10.3390/en13236354.

Chen, J. *et al.* (2020) 'Augmenting a deep-learning algorithm with canal inspection knowledge for reliable water leak detection from multispectral satellite images', *Advanced Engineering Informatics*. Elsevier, 46, p. 101161. doi: 10.1016/j.aei.2020.101161.

Chen, L. *et al.* (2015) 'Multiobjective optimization for maintenance decision making in infrastructure asset management', *Journal of Management in Engineering*, 31(6), p. 04015015. doi: 10.1061/(asce)me.1943-5479.0000371.

Chen, L. and Bai, Q. (2019) 'Optimization in decision making in infrastructure asset management: A review', *Applied Sciences*, 9(7), p. 1380. doi: 10.3390/app9071380.

Chen, Q., Chen, Y. and Jiang, W. (2016) 'Genetic particle swarm optimization–based feature selection for very-high-resolution remotely sensed imagery object change detection', *Sensors (Switzerland)*, 16(8). doi: 10.3390/s16081204.

Chen, S., Leng, Y. and Labi, S. (2020) 'A deep learning algorithm for simulating autonomous driving considering prior knowledge and temporal information', *Computer-Aided Civil and Infrastructure Engineering*, 35(4), pp. 305–321. doi: https://doi.org/10.1111/mice.12495.

Chen, Y. *et al.* (2018) 'Optimal control of HVAC and window systems for natural ventilation through reinforcement learning', *Energy and Buildings*, 169, pp. 195–205. doi: https://doi.org/10.1016/j.enbuild.2018.03.051.

Cheng, M., Yang, D. Y. and Frangopol, D. M. (2020) 'Investigation of effects of time preference and risk perception on life-cycle management of civil infrastructure', *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 6(1), p. 04020001. doi: 10.1061/AJRUA6.0001039.

Chicano, F. *et al.* (2015) 'Fitness probability distribution of Bit-Flip mutation', *Evolutionary computation*, 23(2), pp. 217–248. doi: 10.1162/EVCO_a_00130.

Chollet, F. (2015) *Keras*, *GitHub*. Available at: https://github.com/keras-team/keras (Accessed: 6 January 2021).

Cochrane, E. (2021) 'Senate passes $1 trillion infrastructure bill, handing Biden a bipartisan win'. Available at: https://www.nytimes.com/2021/08/10/us/politics/infrastructure-bill-passes.html.

Dewey, D. (2014) 'Reinforcement learning and the reward engineering principle', in *AAAI Spring Symposia*.

Digalakis, J. G. and Margaritis, K. G. (2001) 'On benchmarking functions for genetic algorithms', *International Journal of Computer Mathematics*, 77(4), pp. 481–506.

doi: 10.1080/00207160108805080.

Dixit, A. K. and Pindyck, R. S. (1994) *Investment under uncertainty*. Princeton university press.

Dong, Y., Frangopol, D. M. and Saydam, D. (2014) 'Pre-earthquake multi-objective probabilistic retrofit optimization of bridge networks based on sustainability', *Journal of Bridge Engineering*, 19(6), p. 4014018. doi: 10.1061/(ASCE)BE.1943-5592.0000586.

Durango-Cohen, P. (2004) 'Maintenance and Repair Decision Making for Infrastructure Facilities without a Deterioration Model', *Journal of Infrastructure Systems - J INFRASTRUCT SYST*, 10. doi: 10.1061/(ASCE)1076-0342(2004)10:1(1).

Edward Best, R. (2016) *Modeling, optimization, and decision support for integrated urban and infrastructure planning*. Stanford University. Available at: https://www.proquest.com/docview/2454193414/977DFBDF0BAB429APQ/1?accountid=16210 (Accessed: 6 January 2021).

Eggertson, L. (2010) 'Lancet retracts 12-year-old article linking autism to MMR vaccines', *CMAJ : Canadian Medical Association journal = journal de l'Association medicale canadienne*. 2010/02/08. Canadian Medical Association, 182(4), pp. E199--E200. doi: 10.1503/cmaj.109-3179.

Elhadidy, A. A., Elbeltagi, E. E. and El-Badawy, S. M. (2020) 'Network-based optimization system for pavement maintenance using a probabilistic simulation-based genetic algorithm approach', *Journal of Transportation Engineering, Part B: Pavements*. American Society of Civil Engineers, 146(4), p. 4020069. doi: 10.1061/JPEODX.0000237.

Ellingham, I. and Fawcett, W. (2007) *New generation whole-life costing: property and construction decision-making under uncertainty*. London and New York: Taylor &

Francis.

Ellingwood, B. R. (2005) 'Risk-informed condition assessment of civil infrastructure: state of practice and research issues', *Structure and Infrastructure Engineering*, 1(1), pp. 7–18. doi: 10.1080/15732470412331289341.

Erharter, G. H. *et al.* (2021) 'Reinforcement learning based process optimization and strategy development in conventional tunneling', *Automation in Construction*. Elsevier B.V., 127, p. 103701. doi: 10.1016/j.autcon.2021.103701.

Evan, M., Nicholas, C. and Sriram, N. (2020) 'Automated defect quantification in concrete bridges using robotics and deep learning', *Journal of Computing in Civil Engineering*. American Society of Civil Engineers, 34(5), p. 4020029. doi: 10.1061/(ASCE)CP.1943-5487.0000915.

FEMA-NIBS: Federal Emergency Management Agency (FEMA) by the National Institute of Building Sciences (2003) *Hazus–MH 2.1: Technical manual*. Washington DC: Federal Emergency Management Agency. Available at: www.fema.gov/plan/prevent/hazus (Accessed: 6 January 2021).

FHWA (Federal Highway Administration) (2012) *Recording and coding guide for the structure inventory and appraisal of the nation's bridges*. Washington, DC: Federal Highway Administration.

Fishburn, P. C. (1968) 'Utility theory', *Management Science*. INFORMS, 14(5), pp. 335–378.

Ford, D. N., Lander, D. M. and Voyer, J. J. (2002) 'A real options approach to valuing strategic flexibility in uncertain construction projects', *Construction Management & Economics*, 20(4), pp. 343–351. doi: 10.1080/01446190210125572.

France-Mensah, J. and O'Brien, W. J. (2018) 'Budget allocation models for pavement maintenance and rehabilitation: comparative case study', *Journal of Management in*

Engineering, 34(2), p. 05018002. doi: 10.1061/(asce)me.1943-5479.0000599.

Frangopol, D. M. (2011) 'Life-cycle performance, management, and optimisation of structural systems under uncertainty: accomplishments and challenges', *Structure and Infrastructure Engineering*, 7(6), pp. 389–413. doi: 10.1080/15732471003594427.

Frangopol, D. M., Dong, Y. and Sabatino, S. (2017) 'Bridge life-cycle performance and cost: analysis, prediction, optimisation and decision-making', *Structure and Infrastructure Engineering*. Taylor & Francis, 13(10), pp. 1239–1257. doi: 10.1080/15732479.2016.1267772.

Gadi, V. K. *et al.* (2020) 'A novel python program to automate soil colour analysis and interpret surface moisture content', *International Journal of Geosynthetics and Ground Engineering*. Springer International Publishing, 6(2), pp. 1–8. doi: 10.1007/s40891-020-00204-3.

Gamma, E. *et al.* (1995) *Design patterns: elements of reusable object-oriented software*. Pearson Education India.

George, L. and George, L. (2018) 'Brownian motion and stochastic processes', in *Energy Power Risk*. Emerald Publishing Limited, pp. 3–32. doi: 10.1108/978-1-78743-527-820181002.

Ghannad, P., Lee, Y. and Choi, J. O. (2021) 'Prioritizing postdisaster recovery of transportation infrastructure systems using multiagent reinforcement learning', *Journal of Management in Engineering*, 37, p. 4020100. doi: 10.1061/(ASCE)ME.1943-5479.0000868.

Ghodoosi, F. *et al.* (2018) 'Maintenance cost optimization for bridge structures using system reliability analysis and genetic algorithms', *Journal of Construction Engineering and Management*, 144(2), p. 04017116. doi: 10.1061/(ASCE)CO.1943-7862.0001435.

Ghosh, J. and Padgett, J. E. (2009) 'Multi-hazard consideration of seismic and aging threats

to bridges', in *Structures Congress 2009: Don't Mess with Structural Engineers: Expanding Our Role*, pp. 1–10. doi: 10.1061/41031(341)61.

Github (2020). Available at: https://github.com (Accessed: 6 January 2021).

Golabi, K., Kulkarni, R. B. and Way, G. B. (1982) 'A statewide pavement management system', *INFORMS Journal on Applied Analytics*, 12(6), pp. 5–21. doi: 10.1287/inte.12.6.5.

Grefenstette, J. J. (1986) 'Optimization of control parameters for genetic algorithms', *IEEE Transactions on systems, man, and cybernetics*, 16(1), pp. 122–128. doi: 10.1109/TSMC.1986.289288.

Guan, X., Burton, H. and Sabol, T. (2020) 'Python-based computational platform to automate seismic design, nonlinear structural model construction and analysis of steel moment resisting frames', *Engineering Structures*. Elsevier, 224, p. 111199. doi: 10.1016/j.engstruct.2020.111199.

Gupta, A. *et al.* (2021) 'Energy-efficient heating control for smart buildings with deep reinforcement learning', *Journal of Building Engineering*, 34, p. 101739. doi: https://doi.org/10.1016/j.jobe.2020.101739.

Haji Hosseinloo, A. *et al.* (2020) 'Data-driven control of micro-climate in buildings: An event-triggered reinforcement learning approach', *Applied Energy*, 277, p. 115451. doi: https://doi.org/10.1016/j.apenergy.2020.115451.

Hall, J. W. *et al.* (2003) 'Integrated flood risk management in England and Wales', *Natural Hazards Review*, 4(3), pp. 126–135. doi: 10.1061/(ASCE)1527-6988(2003)4:3(126).

Hasselt, H. (2010) 'Double Q-learning', in Lafferty, J. et al. (eds) *Advances in Neural Information Processing Systems*. Curran Associates, Inc.

Van Hasselt, H., Guez, A. and Silver, D. (2015) 'Deep Reinforcement Learning with Double Q-learning'.

Hawk, H. and Small, E. P. (1998) 'The BRIDGIT bridge management system', *Structural Engineering International*, 8(4), pp. 309–314. doi: 10.2749/101686698780488712.

Hirsa, A. and Neftci, S. N. (2013) *An introduction to the mathematics of financial derivatives*. Academic press.

Hodge, V. J., Hawkins, R. and Alexander, R. (2021) 'Deep reinforcement learning for drone navigation using sensor data', *Neural Computing and Applications*, 33(6), pp. 2015–2033. doi: 10.1007/s00521-020-05097-x.

Hou, Q. and Ai, C. (2020) 'A network-level sidewalk inventory method using mobile LiDAR and deep learning', *Transportation Research Part C: Emerging Technologies*, 119, p. 102772. doi: https://doi.org/10.1016/j.trc.2020.102772.

Ilbeigi, M. and Ashuri, Baabak Hui, Y. (2014) 'A stochastic process to model the fluctuations of asphalt cement price', in *Construction Research Congress 2014: Construction in a Global Network*.

Ilker, K., D., G. D. and David, J. H. (2020) 'Improving the accuracy of early cost estimates on transportation infrastructure projects', *Journal of Management in Engineering*. American Society of Civil Engineers, 36(5), p. 4020063. doi: 10.1061/(ASCE)ME.1943-5479.0000819.

Jackson, B. *et al.* (2020) 'Characterizing the key predictors of renewable energy penetration for sustainable and resilient communities', *Journal of Management in Engineering*. American Society of Civil Engineers, 36(4), p. 4020016. doi: 10.1061/(ASCE)ME.1943-5479.0000767.

Jafari, A. and Valentin, V. (2017) 'An optimization framework for building energy retrofits decision-making', *Building and Environment*, 115, pp. 118–129. doi: 10.1016/j.buildenv.2017.01.020.

Jagadale, U. T. *et al.* (2020) 'An experimental-based python programming for structural

health monitoring of non-engineered RC frame', *Innovative Infrastructure Solutions*. Springer International Publishing, 5(1), pp. 1–10. doi: 10.1007/s41062-020-0260-x.

Jeong, Y. *et al.* (2018) 'Bridge inspection practices and bridge management programs in China, Japan, Korea, and U.S.', *Journal of Structural Integrity and Maintenance*. Taylor & Francis, 3(2), pp. 126–135. doi: 10.1080/24705314.2018.1461548.

Jiang, R., Wu, P. and Wu, C. (2021) 'Selecting the optimal network-level pavement maintenance budget scenario based on sustainable considerations', *Transportation Research Part D: Transport and Environment*, 97, p. 102919. doi: https://doi.org/10.1016/j.trd.2021.102919.

Jin, W. and Jiansong, Z. (2019) 'New automated BIM object classification method to support BIM interoperability', *Journal of Computing in Civil Engineering*. American Society of Civil Engineers, 33(5), p. 4019033. doi: 10.1061/(ASCE)CP.1943-5487.0000858.

Jung, S. *et al.* (2021) 'Optimal planning of a rooftop PV system using GIS-based reinforcement learning', *Applied Energy*, 298, p. 117239. doi: https://doi.org/10.1016/j.apenergy.2021.117239.

Kaganova, O. and Telgarsky, J. (2018) 'Management of capital assets by local governments: An assessment and benchmarking survey', *International Journal of Strategic Property Management*, 22(2), pp. 143–156. doi: 10.3846/ijspm.2018.445.

Kandil, A., El-Rayes, K. and El-Anwar, O. (2010) 'Optimization research: Enhancing the robustness of large-scale multiobjective optimization in construction', *Journal of Construction Engineering and Management*, 136(1), pp. 17–25. doi: 10.1061/(ASCE)CO.1943-7862.0000140.

Keskar, N. S. *et al.* (2017) 'On large-batch training for deep learning: Generalization gap and sharp minima', in *International Conference on Learning Representations*. doi: arXiv:1609.04836.

Khadilkar, H. (2019) ‘A scalable reinforcement learning algorithm for scheduling railway lines’, *IEEE Transactions on Intelligent Transportation Systems*, 20(2), pp. 727–736. doi: 10.1109/TITS.2018.2829165.

Khalifa, N. A. *et al.* (2019) ‘Critical review of flexible pavement maintenance management systems’, *International Journal of Engineering & Technology*, 8(2), pp. 95–101. doi: 10.13140/RG.2.2.18001.68967.

Khazaeli, S., Nguyen, L. H. and Goulet, J. A. (2021) ‘Anomaly detection using state-space models and reinforcement learning’, *Structural Control and Health Monitoring*. John Wiley & Sons, Ltd, 28(6), p. e2720. doi: https://doi.org/10.1002/stc.2720.

Khouzani, A. H. E., Golroo, A. and Bagheri, M. (2017) ‘Railway maintenance management using a stochastic geometrical degradation model’, *Journal of Transportation Engineering, Part A: Systems*, 143(1), p. 4016002. doi: 10.1061/JTEPBS.0000002.

Kim, S. and Frangopol, D. M. (2018) ‘Decision making for probabilistic fatigue inspection planning based on multi-objective optimization’, *International Journal of Fatigue*. Elsevier, 111, pp. 356–368. doi: 10.1016/j.ijfatigue.2018.01.027.

Kim, Y. *et al.* (2017) ‘Probabilistic cash flow-based optimal investment timing using two-color rainbow options valuation for economic sustainability appraisement’, *Sustainability*, 9(10), p. 1781. doi: 10.3390/su9101781.

Kim, Y. and Lee, E. B. (2018) ‘Optimal investment timing with investment propensity using fuzzy real options valuation’, *International Journal of Fuzzy Systems*. Springer Berlin Heidelberg, 20(6), pp. 1888–1900. doi: 10.1007/s40815-018-0493-4.

Kingma, D. P. and Ba, J. (2015) ‘Adam: A method for stochastic optimization’, in *3rd International Conference for Learning Representations, San Diego*.

Kirkpatrick, J. *et al.* (2017) ‘Overcoming catastrophic forgetting in neural networks’, *Proceedings of the National Academy of Sciences*, 114(13), pp. 3521 LP – 3526. doi:

10.1073/pnas.1611835114.

Klerk, W. J. *et al.* (2019) 'Value of information of structural health monitoring in asset management of flood defences', *Infrastructures*, 4(3), pp. 1–20. doi: 10.3390/infrastructures4030056.

Konda, V. and Tsitsiklis, J. (2001) 'Actor-critic algorithms', *Society for Industrial and Applied Mathematics*, 42.

Krishna Lakshmanan, A. *et al.* (2020) 'Complete coverage path planning using reinforcement learning for Tetromino based cleaning and maintenance robot', *Automation in Construction*. Elsevier B.V., 112, p. 103078. doi: 10.1016/j.autcon.2020.103078.

Lagaros, N. D., Kepaptsoglou, K. and Karlaftis, M. G. (2013) 'Fund allocation for civil infrastructure security upgrade', *Journal of Management in Engineering*, 29(2), pp. 172–182. doi: 10.1061/(asce)me.1943-5479.0000133.

Lamptey, G., Labi, S. and Li, Z. (2008) 'Decision support for optimal scheduling of highway pavement preventive maintenance within resurfacing cycle', *Decision Support Systems*. Elsevier B.V., 46(1), pp. 376–387. doi: 10.1016/j.dss.2008.07.004.

Law, A. M. and Kelton, W. D. (2000) *Simulation modeling and analysis*. 3rd edn. New York: McGraw-Hill.

Lee, D. and Kim, M. (2021) 'Autonomous construction hoist system based on deep reinforcement learning in high-rise building construction', *Automation in Construction*, 128, p. 103737. doi: https://doi.org/10.1016/j.autcon.2021.103737.

Levine, S. S. and Prietula, M. J. (2014) 'Open collaboration for innovation: principles and performance', *Organization Science*, 25(5), pp. 1414–1433. doi: 10.1287/orsc.2013.0872.

Li, Y. *et al.* (2020) 'Long-term resilience and loss assessment of highway bridges under multiple natural hazards', *Structure and Infrastructure Engineering*, 16(4), pp. 626–

641. doi: 10.1080/15732479.2019.1699936.

Li, Z. and Sinha, K. C. (2004) 'Methodology for multicriteria decision making in highway asset management', *Transportation Research Record*, 1885, pp. 79–87. doi: 10.3141/1885-12.

Liang, C. J., Kamat, V. R. and Menassa, C. C. (2020) 'Teaching robots to perform quasi-repetitive construction tasks through human demonstration', *Automation in Construction*. Elsevier B.V., 120, p. 103370. doi: 10.1016/j.autcon.2020.103370.

Lissa, P. *et al.* (2021) 'Deep reinforcement learning for home energy management system control', *Energy and AI*, 3, p. 100043. doi: https://doi.org/10.1016/j.egyai.2020.100043.

Liu, D., Cao, J. and Lei, X. (2019) 'Slabstone Installation Skill Acquisition for Dual-Arm Robot based on Reinforcement Learning', in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1298–1305. doi: 10.1109/ROBIO49542.2019.8961805.

Liu, J. *et al.* (2020) 'Towards automated clash resolution of reinforcing steel design in reinforced concrete frames via Q-learning and building information modeling', *Automation in Construction*. Elsevier B.V., 112, p. 103062. doi: 10.1016/j.autcon.2019.103062.

Liu, L. *et al.* (2018) 'Sustainability-informed bridge ranking under scour based on transportation network performance and multiattribute utility', *Journal of Bridge Engineering*, 23(10), p. 04018082. doi: 10.1061/(ASCE)BE.1943-5592.0001296.

Liu, S. and Henze, G. P. (2006) 'Evaluation of Reinforcement Learning for Optimal Control of Building Active and Passive Thermal Storage Inventory', *Journal of Solar Energy Engineering*, 129(2), pp. 215–225. doi: 10.1115/1.2710491.

Lounis, Z. and McAllister, T. P. (2016) 'Risk-based decision making for sustainable and

resilient infrastructure systems', *Journal of Structural Engineering*, 142(9), p. F4016005. doi: 10.1061/(ASCE)ST.1943-541X.0001545.

Lu, Z., Xuan, L. and Sunil, D. (2021) 'A reinforcement learning-based stakeholder value aggregation model for collaborative decision making on disaster resilience', *Computing in Civil Engineering 2019*. (Proceedings), pp. 490–497. doi: doi:10.1061/9780784482445.063.

M., C. C., Oscar, O. and Jeffrey, W. (2021) 'Integrating the risk of climate change into transportation asset management to support bridge network-level decision-making', *Journal of Infrastructure Systems*. American Society of Civil Engineers, 27(1), p. 4020044. doi: 10.1061/(ASCE)IS.1943-555X.0000590.

Mahsuli, M. and Haukaas, T. (2013) 'Computer program for multimodel reliability and optimization analysis', *Journal of Computing in Civil Engineering*, 27(1), pp. 87–98. doi: 10.1061/(asce)cp.1943-5487.0000204.

Mandow, L. *et al.* (2020) 'Architectural planning with shape grammars and reinforcement learning: Habitability and energy efficiency', *Engineering Applications of Artificial Intelligence*, 96, p. 103909. doi: https://doi.org/10.1016/j.engappai.2020.103909.

McDonnell, B. E. *et al.* (2020) 'PySWMM: The Python interface to stormwater management model (SWMM)', *Journal of Open Source Software*, 5(52), p. 2292. doi: 10.21105/joss.02292.

McGuire, R. (2004) *Seismic hazard and risk analysis*. Berkeley, CA: Earthquake Engineering Research Institute.

Memarzadeh, M. and Pozzi, M. (2019) 'Model-free reinforcement learning with model-based safe exploration: Optimizing adaptive recovery process of infrastructure systems', *Structural Safety*, 80, pp. 46–55. doi: https://doi.org/10.1016/j.strusafe.2019.04.003.

Miyamoto, A., Kawamura, K. and Nakamura, H. (2000) 'Bridge management system and

maintenance optimization for existing bridges', *Computer-Aided Civil and Infrastructure Engineering*, 15(1), pp. 45–55. doi: 10.1111/0885-9507.00170.

Mnih, V. *et al.* (2013) 'Playing Atari with Deep Reinforcement Learning', *arXiv preprint arXiv: 1312.5602*. Available at: http://arxiv.org/abs/1312.5602.

Mnih, V. *et al.* (2016) 'Asynchronous methods for deep reinforcement learning', in Balcan, M. F. and Q, W. K. (eds) *The 33rd International Conference on Machine Learning (ICML 2016)*. PMLR, pp. 1928–1937. Available at: http://proceedings.mlr.press/v48/mniha16.pdf.

Mohamad, A., Jordan, S. F. and M., S. I. (2021) 'Data-driven machine learning approach to integrate field submittals in project scheduling', *Journal of Management in Engineering*. American Society of Civil Engineers, 37(1), p. 4020104. doi: 10.1061/(ASCE)ME.1943-5479.0000873.

Mondoro, A., Frangopol, D. M. and Liu, L. (2018) 'Multi-criteria robust optimization framework for bridge adaptation under climate change', *Structural Safety*, 74, pp. 14–23. doi: https://doi.org/10.1016/j.strusafe.2018.03.002.

Montazeri, N. and Touran, A. (2019) 'Applied decision-making framework for maintenance scheduling in bridge management', in *Precedings on the Creative Construction Conference*, pp. 547–555. doi: 10.3311/ccc2019-075.

Mullapudi, A. *et al.* (2020) 'Deep reinforcement learning for the real time control of stormwater systems', *Advances in Water Resources*, 140, p. 103600. doi: https://doi.org/10.1016/j.advwatres.2020.103600.

Murugesan, S. *et al.* (2020) 'Less is more: Simplified state-action space for deep reinforcement learning based HVAC control', in *1st International Workshop on Reinforcement Learning for Energy Management in Buildings and Cities, RLEM 2020*. AI Hub, Johnson Controls, Inc., Santa Clara, CA, United States: Association for

Computing Machinery, Inc, pp. 20–23. doi: 10.1145/3427773.3427864.

Mushtaq, A. *et al.* (2021) 'Traffic flow management of autonomous vehicles using deep reinforcement learning and smart rerouting', *IEEE Access*, 9, pp. 51005–51019. doi: 10.1109/ACCESS.2021.3063463.

Ng, A. (2018) *Machine Learning Yearning*. Available at: https://www.deeplearning.ai/programs/ (Accessed: 28 October 2021).

Nguyen, H. and La, H. (2019) 'Review of Deep Reinforcement Learning for Robot Manipulation', in *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pp. 590–595. doi: 10.1109/IRC.2019.00120.

Nguyen, T. T., Nguyen, N. D. and Nahavandi, S. (2020) 'Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications', *IEEE Transactions on Cybernetics*, 50(9), pp. 3826–3839. doi: 10.1109/TCYB.2020.2977374.

Nian, R., Liu, J. and Huang, B. (2020) 'A review on reinforcement learning: Introduction and applications in industrial process control', *Computers & Chemical Engineering*, 139, p. 106886. doi: 10.1016/j.compchemeng.2020.106886.

Nicholson, W. and Christopher, S. (2011) *Microeconomic theory: Basic principles and extensions*. South-Western College Pub.

Nozhati, S., Ellingwood, B. R. and Chong, E. K. P. (2020) 'Stochastic optimal control methodologies in risk-informed community resilience planning', *Structural Safety*, 84, p. 101920. doi: https://doi.org/10.1016/j.strusafe.2019.101920.

Orcesi, A. D. and Frangopol, D. M. (2011) 'Optimization of bridge maintenance strategies based on structural health monitoring information', *Structural Safety*, 33(1), pp. 26–41. doi: 10.1016/j.strusafe.2010.05.002.

Ou, X., Chang, Q. and Chakraborty, N. (2021) 'A method integrating Q-learning with

approximate dynamic programming for gantry work cell scheduling', *IEEE Transactions on Automation Science and Engineering*, 18(1), pp. 85–93. doi: 10.1109/TASE.2020.2984739.

Pagani, M. *et al.* (2014) 'OpenQuake engine: An open hazard (and risk) software for the global earthquake model', *Seismological Research Letters*, 85(3), pp. 692–702. doi: 10.1785/0220130087.

Papakonstantinou, K. G. and Shinozuka, M. (2014) 'Planning structural inspection and maintenance policies via dynamic programming and Markov processes. Part II: POMDP implementation', *Reliability Engineering & System Safety*, 130, pp. 214–224. doi: https://doi.org/10.1016/j.ress.2014.04.006.

Park, J. Y. *et al.* (2019) 'LightLearn: An adaptive and occupant centered controller for lighting based on reinforcement learning', *Building and Environment*. Elsevier Ltd, 147, pp. 397–414. doi: 10.1016/j.buildenv.2018.10.028.

Patidar, V., Labi, S., Morin, T., Thompson, Paul D., *et al.* (2011) 'Evaluating methods and algorithms for multicriteria bridge management at the network level', *Transportation Research Record*, 2220(1), pp. 38–47. doi: 10.3141/2220-05.

Patidar, V., Labi, S., Morin, T., Thompson, Paul D, *et al.* (2011) 'Evaluating methods and algorithms for multicriteria bridge management at the network level', *Transportation Research Record*, 2220(1), pp. 38–47. doi: 10.3141/2220-05.

Pedregosa, F. *et al.* (2011) 'Scikit-learn: machine learning in Python', *Journal of Machine Learning Research*, 12, pp. 2825–2830. Available at: http://jmlr.org/papers/v12/pedregosa11a.html.

Peraka, N. S. P. and Biligiri, K. P. (2020) 'Pavement asset management systems and technologies: A review', *Automation in Construction*, 119, p. 103336. doi: https://doi.org/10.1016/j.autcon.2020.103336.

Pindyck, R. S. (1993) 'Investments of uncertain cost', *Journal of Financial Economics*, 34(1), pp. 53–76. doi: 10.1016/0304-405X(93)90040-I.

Pouya, Z., Hesam, H. and Brenda, M. (2020) 'Quantifying remoteness for risk and resilience assessment using nighttime satellite imagery', *Journal of Computing in Civil Engineering*. American Society of Civil Engineers, 34(5), p. 4020026. doi: 10.1061/(ASCE)CP.1943-5487.0000906.

Rahimi, H. and Mahsuli, M. (2019) 'Structural reliability approach to analysis of probabilistic seismic hazard and its sensitivities', *Bulletin of Earthquake Engineering*. Springer Netherlands, 17(3), pp. 1331–1359. doi: 10.1007/s10518-018-0497-3.

Rasheed, F. *et al.* (2020) 'Deep reinforcement learning for traffic signal control: A review', *IEEE Access*, 8, pp. 208016–208044. doi: 10.1109/ACCESS.2020.3034141.

Raso, L. *et al.* (2016) 'Optimist: A python library for water system optimal operation and analysis using SDDP', in *International Congress on Environmental Modelling and Software, iEMSs*, pp. 926–932. Available at: https://scholarsarchive.byu.edu/iemssconference/2016/Stream-D/26/ (Accessed: 6 January 2020).

Rayan, A. and H., E. I. (2020) 'Evaluation and prediction of the hazard potential level of dam infrastructures using computational artificial intelligence algorithms', *Journal of Management in Engineering*. American Society of Civil Engineers, 36(5), p. 4020051. doi: 10.1061/(ASCE)ME.1943-5479.0000810.

Redmon, J. *et al.* (2016) 'You only look once: Unified, real-time object detection', in *Proceedings of the IEEE conference on computer vision and pattern recognition.*

Renard, S., Corbett, B. and Swei, O. (2021) 'Minimizing the global warming impact of pavement infrastructure through reinforcement learning', *Resources, Conservation and Recycling*, 167, p. 105240. doi: https://doi.org/10.1016/j.resconrec.2020.105240.

Renna, P. (2017) ‘A decision investment model to design manufacturing systems based on a genetic algorithm and Monte-Carlo simulation’, *International Journal of Computer Integrated Manufacturing*. Taylor & Francis, 30(6), pp. 590–605. doi: 10.1080/0951192X.2016.1187299.

Rivera S, J. A. and Estupiñán L, A. F. (2020) ‘Development of a computational software in Python, used to study the materials resistance in beams’, *arXiv e-prints*, p. arXiv:2009.09448v2. Available at: http://arxiv.org/abs/2009.09448v2 (Accessed: 6 January 2021).

Ross, S. M. (2010) *Introduction to probability models*. 10th edn. Academic Press.

Ruiz-Montiel, M. *et al.* (2013) ‘Design with shape grammars and reinforcement learning’, *Advanced Engineering Informatics*, 27(2), pp. 230–245. doi: https://doi.org/10.1016/j.aei.2012.12.004.

Rummery, G. A. and Niranjan, M. (1994) ‘On-line Q-learning using connectionist systems’, in.

Sahachaisaree, S., Sae-ma, P. and Nanakorn, P. (2020) ‘Two-dimensional truss topology design by reinforcement learning’, in, pp. 1237–1245. doi: 10.1007/978-981-15-5144-4_122.

Sai, P. G., Sanayei, M. and Smith, I. F. (2021) ‘Model-class selection using clustering and classification for structural identification and prediction’, *Journal of Computing in Civil Engineering*. American Society of Civil Engineers, 35(1), p. 4020051. doi: 10.1061/(ASCE)CP.1943-5487.0000932.

Salimi, S., Mawlana, M. and Hammad, A. (2014) ‘Simulation-based multiobjective optimization of bridge construction processes using parallel computing’, in *Proceedings of the 2014 Winter Simulation Conference*, pp. 3272–3283.

Sánchez-Silva, M. *et al.* (2016) ‘Maintenance and operation of infrastructure systems:

Review', *Journal of Structural Engineering*, 142(9), pp. 1–16. doi: 10.1061/(ASCE)ST.1943-541X.0001543.

Santos, J., Ferreira, A. and Flintsch, G. (2019) 'An adaptive hybrid genetic algorithm for pavement management', *International Journal of Pavement Engineering*. Taylor & Francis, 20(3), pp. 266–286. doi: 10.1080/10298436.2017.1293260.

Saydam, D. and Frangopol, D. M. (2015) 'Risk-based maintenance optimization of deteriorating bridges', *Journal of Structural Engineering (United States)*, 141(4), pp. 1–10. doi: 10.1061/(ASCE)ST.1943-541X.0001038.

Schaul, T. *et al.* (2016) 'Prioritized experience replay', *CoRR*, abs/1511.0.

Serrano, W. (2019) 'Deep reinforcement learning algorithms in intelligent infrastructure', *Infrastructures*, 4(3). doi: 10.3390/infrastructures4030052.

Shafiee, M. and Sørensen, J. D. (2019) 'Maintenance optimization and inspection planning of wind energy assets: Models, methods and strategies', *Reliability Engineering & System Safety*, 192, p. 105993. doi: 10.1016/j.ress.2017.10.025.

Shi, F. *et al.* (2020) 'Addressing adjacency constraints in rectangular floor plans using Monte-Carlo Tree Search', *Automation in Construction*. Elsevier B.V., 115, p. 103187. doi: 10.1016/j.autcon.2020.103187.

Shi, Y. *et al.* (2019) 'Impact assessment of reinforced learning methods on construction workers' fall risk behavior using virtual reality', *Automation in Construction*. Elsevier B.V., 104, pp. 197–214. doi: 10.1016/j.autcon.2019.04.015.

Shin, J. *et al.* (2019) 'Reinforcement Learning – Overview of recent progress and implications for process control', *Computers & Chemical Engineering*, 127, pp. 282–294. doi: https://doi.org/10.1016/j.compchemeng.2019.05.029.

Si, C. *et al.* (2021) 'Deep reinforcement learning based home energy management system with devices operational dependencies', *International Journal of Machine Learning*

and Cybernetics, 12(6), pp. 1687–1703. doi: 10.1007/s13042-020-01266-5.

Silver, D. *et al.* (2014) 'Deterministic policy gradient algorithms', in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*. JMLR.org (ICML'14), pp. I--387--I--395.

Silver, D. *et al.* (2016) 'Mastering the game of Go with deep neural networks and tree search', *Nature*. Nature Publishing Group, 529(7587), pp. 484–489. doi: 10.1038/nature16961.

Sinha, K. C. *et al.* (2009) *Updating and enhancing the Indiana bridge management system (IBMS)*, *Publication FHWA/IN/JTRP-2008/30. Joint Transportation Research Program, Indiana Department of Transportation and Purdue University*. West Lafayette, Indiana,. doi: 10.5703/1288284314306.

Socher, R. *et al.* (2011) 'Parsing natural scenes and natural language with recursive neural networks', *Proceedings of the 28th international conference on machine learning (ICML-11)*, 5781, pp. 129–146. doi: 10.1007/978-3-642-04180-8.

Soliman, A. A. S., Ahmed, M. and Tarek, Z. (2017) 'Decision-making framework for integrated asset management', in *Proceedings, Annual Conference - Canadian Society for Civil Engineering*, pp. 161–170.

Straub, D. and Faber, M. H. (2005) 'Risk based inspection planning for structural systems', *Structural Safety*, 27(4), pp. 335–355. doi: 10.1016/j.strusafe.2005.04.001.

Sun, Z. *et al.* (2019) 'Bayesian network modeling of airport runway incursion occurring processes for predictive accident control', in *Advances in Informatics and Computing in Civil and Construction Engineering: Proceeding of the 35th CIB W78 2018 Conference - IT in Design, Construction, and Management*. Chicago: CIB, pp. 669–676. doi: https://doi.org/10.1007/978-3-030-00220-6_80.

Sun, Z. *et al.* (2020) 'Predictive nuclear power plant outage control through computer vision

and data-driven simulation', *Progress in Nuclear Energy*. Elsevier Ltd, 127, p. 103448. doi: 10.1016/j.pnucene.2020.103448.

Sutton, R. S. *et al.* (1999) 'Policy gradient methods for reinforcement learning with function approximation', in *Proceedings of the 12th International Conference on Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press (NIPS'99), pp. 1057–1063.

Sutton, R. S. and Barto, A. G. (2018) *Reinforcement learning: An introduction*. Cambridge, MA, USA: A Bradford Book.

Tabandeh, A. and Gardoni, P. (2015) 'Empirical bayes approach for developing hierarchical probabilistic predictive models and its application to the seismic reliability analysis of FRP-retrofitted RC bridges', *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 1(2), pp. 1–21. doi: 10.1061/AJRUA6.0000817.

Talebiyan, H. and Mahsuli, M. (2018) 'Risk-based prioritization of a building portfolio for retrofit', *Journal of Structural Engineering*, 144(1), p. 04017181. doi: 10.1061/(ASCE)ST.1943-541X.0001927.

Taneja, P. *et al.* (2010) 'Real options for port infrastructure investments', in *Next Generation Infrastructure Systems for Eco-Cities*, pp. 1–6. doi: 10.1109/INFRA.2010.5679225.

TexasDOT (2020) *Road user costs*. Available at: https://www.txdot.gov/inside-txdot/division/construction/road-user-costs.html (Accessed: 6 January 2021).

Thompson, P. D. *et al.* (1998) 'The Pontis bridge management system', *Structural engineering international*, 8(4), pp. 303–308. doi: 10.2749/101686698780488758.

Thompson, P. D. *et al.* (2016) 'Geotechnical asset management plan: Analysis of life-cycle cost and risk', *Transportation Research Record*, 2596(1), pp. 36–43. doi: 10.3141/2596-05.

Thöns, S. (2018) 'On the value of monitoring information for the structural integrity and risk management', *Computer-Aided Civil and Infrastructure Engineering*, 33(1), pp. 79–94. doi: 10.1111/mice.12332.

Torres-Machi, C. *et al.* (2017) 'Towards a sustainable optimization of pavement maintenance programs under budgetary restrictions', *Journal of Cleaner Production*, 148, pp. 90–102. doi: 10.1016/j.jclepro.2017.01.100.

Torvalds, L. (2020) *Linux [Operating System]*, *GitHub*. Available at: https://github.com/torvalds/linux (Accessed: 6 January 2021).

USGS (2020) *United States Geological Survey*. Available at: https://earthquake.usgs.gov/earthquakes/search/ (Accessed: 6 January 2021).

Valladares, W. *et al.* (2019) 'Energy optimization associated with thermal comfort and indoor air control via a deep reinforcement learning algorithm', *Building and Environment*, 155, pp. 105–117. doi: https://doi.org/10.1016/j.buildenv.2019.03.038.

Vázquez-Canteli, J. R. and Nagy, Z. (2019) 'Reinforcement learning for demand response: A review of algorithms and modeling techniques', *Applied Energy*, 235, pp. 1072–1089. doi: https://doi.org/10.1016/j.apenergy.2018.11.002.

W. Ryan, T. *et al.* (2012) *Bridge inspector's reference manual*. FHWA NHI 1. FHWA (Federal highway association). Available at: http://tinyurl.com/y3v5gq9v.

Wang, C. *et al.* (2020) 'MDP-based distribution network reconfiguration with renewable distributed generation: Approximate dynamic programming approach', *IEEE Transactions on Smart Grid*, 11(4), pp. 3620–3631. doi: 10.1109/TSG.2019.2963696.

Wang, R. *et al.* (2020) 'Detecting corporate misconduct through random forest in China's construction industry', *Journal of Cleaner Production*. Elsevier Ltd, 268, p. 122266. doi: 10.1016/j.jclepro.2020.122266.

Wang, Y. and Tang, P. (2021) 'Spatiotemporal data-driven simulation and clustering of

ground operations of aircraft for comprehending airport jams and collisions', *Construction Research Congress 2020*. (Proceedings), pp. 982–991. doi: doi:10.1061/9780784482865.104.

Wang, Z., Freitas, N. and Lanctot, M. (2015) 'Dueling network architectures for deep reinforcement learning'.

Warren, C., Giannopoulos, A. and Giannakis, I. (2016) 'gprMax: Open source software to simulate electromagnetic wave propagation for Ground Penetrating Radar', *Computer Physics Communications*. Elsevier B.V., 209, pp. 163–170. doi: 10.1016/j.cpc.2016.08.020.

Watkins, C. J. C. H. and Dayan, P. (1992) 'Q-learning', *Machine Learning*, 8(3), pp. 279–292. doi: 10.1007/BF00992698.

Wei, P. *et al.* (2020) 'A deep-reinforcement-learning-based recommender system for occupant-driven energy optimization in commercial buildings', *IEEE Internet of Things Journal*, 7(7), pp. 6402–6413. doi: 10.1109/JIOT.2020.2974848.

Wei, S., Bao, Y. and Li, H. (2020) 'Optimal policy for structure maintenance: A deep reinforcement learning framework', *Structural Safety*, 83, p. 101906. doi: https://doi.org/10.1016/j.strusafe.2019.101906.

Wei, T., Wang, Y. and Zhu, Q. (2017) 'Deep reinforcement learning for building HVAC control', in *2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6. doi: 10.1145/3061639.3062224.

Whitley, D. (1994) 'A genetic algorithm tutorial', *Statistics and computing*, 4(2), pp. 65–85. doi: 10.1007/BF00175354.

Witt, C. (2013) 'Tight bounds on the optimization time of a randomized search heuristic on linear functions', *Combinatorics, Probability & Computing*, 22(2), pp. 294–318. doi: 10.1017/S0963548312000600.

Xia, K. *et al.* (2021) 'A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence', *Journal of Manufacturing Systems*, 58, pp. 210–230. doi: https://doi.org/10.1016/j.jmsy.2020.06.012.

Xiang, Z. *et al.* (2020) 'Deep reinforcement learning-based sampling method for structural reliability assessment', *Reliability Engineering & System Safety*, 199, p. 106901. doi: https://doi.org/10.1016/j.ress.2020.106901.

Xu, B. *et al.* (2015) 'Empirical evaluation of rectified activations in convolutional network', *arXiv:1505.00853*. Available at: http://arxiv.org/abs/1505.00853.

Xu, Y. *et al.* (2021) 'Machine learning in construction: From shallow to deep learning', *Developments in the Built Environment*. Elsevier Ltd, 6(April 2020), p. 100045. doi: 10.1016/j.dibe.2021.100045.

Xuan, P. *et al.* (2011) 'Genetic algorithm-based efficient feature selection for classification of pre-miRNAs', *Genetics and Molecular Research*, 10(2), pp. 588–603. doi: 10.4238/vol10-2gmr969.

Yang, D. Y. and Frangopol, D. M. (2019) 'Life-cycle management of deteriorating civil infrastructure considering resilience to lifetime hazards: A general approach based on renewal-reward processes', *Reliability Engineering & System Safety*, 183, pp. 197–212. doi: https://doi.org/10.1016/j.ress.2018.11.016.

Yang, David Y. and Frangopol, D. M. (2020a) 'Life-cycle management of deteriorating bridge networks with network-level risk bounds and system reliability analysis', *Structural Safety*, 83, p. 101911. doi: 10.1016/j.strusafe.2019.101911.

Yang, David Y. and Frangopol, D. M. (2020b) 'Risk-based portfolio management of civil infrastructure assets under deep uncertainties associated with climate change: a robust optimisation approach', *Structure and Infrastructure Engineering*. Taylor & Francis,

16(4), pp. 531–546. doi: 10.1080/15732479.2019.1639776.

Yang, David Y and Frangopol, D. M. (2020) 'Risk-based portfolio management of civil infrastructure assets under deep uncertainties associated with climate change: a robust optimisation approach', *Structure and Infrastructure Engineering*. Taylor & Francis, 16(4), pp. 531–546. doi: 10.1080/15732479.2019.1639776.

Yang, D. Y., Frangopol, D. M. and Teng, J. G. (2019) 'Probabilistic life-cycle optimization of durability-enhancing maintenance actions: Application to FRP strengthening planning', *Engineering Structures*, 188, pp. 340–349. doi: 10.1016/j.engstruct.2019.02.055.

Yang, I. T., Hsieh, Y. M. and Kung, L. O. (2012) 'Parallel computing platform for multiobjective simulation optimization of bridge maintenance planning', *Journal of Construction Engineering and Management*, 138(2), pp. 215–226. doi: 10.1061/(ASCE)CO.1943-7862.0000421.

Yang, L. *et al.* (2015) 'Reinforcement learning for optimal control of low exergy buildings', *Applied Energy*, 156, pp. 577–586. doi: https://doi.org/10.1016/j.apenergy.2015.07.050.

Yang, Y. *et al.* (2019) 'Towards resilient civil infrastructure asset management: An information elicitation and analytical framework', *Sustainability*, 11(16), p. 4439. doi: 10.3390/su11164439.

Yao, L. *et al.* (2020) 'Deep reinforcement learning for long-term pavement maintenance planning', *Computer-Aided Civil and Infrastructure Engineering*. John Wiley & Sons, Ltd, 35(11), pp. 1230–1245. doi: https://doi.org/10.1111/mice.12558.

Ye, H., Li, G. Y. and Juang, B.-H. F. (2019) 'Deep reinforcement learning based resource allocation for V2V communications', *IEEE Transactions on Vehicular Technology*, 68(4), pp. 3163–3173. doi: 10.1109/TVT.2019.2897134.

Ye, Y. *et al.* (2021) 'Real-time autonomous residential demand response management based on twin delayed deep deterministic policy gradient learning', *Energies*, 14, p. 531. doi: 10.3390/en14030531.

Yehia, A. (2020) 'Understanding uncertainty : a reinforcement learning approach for project-level pavement management systems'. Available at: https://open.library.ubc.ca/collections/24/items/1.0389993.

Yudi, C., Qi, W. and Wenying, J. (2020) 'Rapid assessment of disaster impacts on highways using social media', *Journal of Management in Engineering*. American Society of Civil Engineers, 36(5), p. 4020068. doi: 10.1061/(ASCE)ME.1943-5479.0000836.

Zeraoui, A. *et al.* (2020) 'New software for the optimization of the formulation and the treatment of dredged sediments for utilization in civil engineering', *Journal of Soils and Sediments*. Journal of Soils and Sediments, 20, pp. 2709–2716. doi: 10.1007/s11368-020-02605-3.

Zhang, K., Yang, Z. and Başar, T. (2021) 'Multi-agent reinforcement learning: A selective overview of theories and algorithms BT - handbook of reinforcement learning and control', in Vamvoudakis, K. G. et al. (eds). Cham: Springer International Publishing, pp. 321–384. doi: 10.1007/978-3-030-60990-0_12.

Zhang, S., Yao, H. and Whiteson, S. (2021) *Breaking the Deadly Triad with a Target Network*.

Zhang, Y. *et al.* (2017) 'Optimal sustainable life cycle maintenance strategies for port infrastructures', *Journal of Cleaner Production*, 142, pp. 1693–1709. doi: 10.1016/j.jclepro.2016.11.120.

Zhang, Z. *et al.* (2019) 'Whole building energy model for HVAC optimal control: A practical framework based on deep reinforcement learning', *Energy and Buildings*, 199, pp. 472–490. doi: https://doi.org/10.1016/j.enbuild.2019.07.029.

Zhong, C., Ng, S. T. and Skitmore, M. (2019) 'An interdependent infrastructure asset management framework for high-density cities', in *Proceedings of the Institution of Civil Engineers - Municipal Engineer*, pp. 1–11. doi: 10.1680/jmuen.18.00053.

Zhou, J. *et al.* (2019) 'Random forests and cubist algorithms for predicting shear strengths of rockfill materials', *Applied Sciences*, p. 1621. doi: 10.3390/app9081621.

Zhu, M., McKenna, F. and Scott, M. H. (2018) 'OpenSeesPy: Python library for the OpenSees finite element framework', *SoftwareX*. Elsevier B.V., 7, pp. 6–11. doi: 10.1016/j.softx.2017.10.009.

Zou, Z., Yu, X. and Ergan, S. (2020) 'Towards optimal control of air handling units using deep reinforcement learning and recurrent neural network', *Building and Environment*. Elsevier Ltd, 168, p. 106535. doi: 10.1016/j.buildenv.2019.106535.