8-2022

Optimizing Strategic Planning With Long-term Sequential Decision Making Under Uncertainty: A Decomposition Approach

Zeyu Liu
University of Tennessee, Knoxville, zliu65@vols.utk.edu

Recommended Citation

Liu, Zeyu, "Optimizing Strategic Planning With Long-term Sequential Decision Making Under Uncertainty: A Decomposition Approach. " PhD diss., University of Tennessee, 2022.
https://trace.tennessee.edu/utk_graddiss/7246

To the Graduate Council:

I am submitting herewith a dissertation written by Zeyu Liu entitled "Optimizing Strategic Planning With Long-term Sequential Decision Making Under Uncertainty: A Decomposition Approach." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Industrial Engineering.

Xueping Li and Anahita Khojandi, Major Professor

We have read this dissertation and recommend its acceptance:

Xueping Li, Anahita Khojandi, Olufemi Omitaomu, Shuai Li

Accepted for the Council:

Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

I am submitting herewith a thesis written by Zeyu Liu entitled "Optimizing Strategic Planning With Long-term Sequential Decision Making Under Uncertainty: A Decomposition Approach." I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Industrial Engineering.

Xueping Li and Anahita Khojandi, Major Professors

We have read this thesis
and recommend its acceptance:

Xueping Li

Anahita Khojandi

Olufemi Omitaomu

Shuai Li

Accepted for the Council:

Dixie Thompson

Vice Provost and Dean of the Graduate School

To the Graduate Council:
I am submitting herewith a thesis written by Zeyu Liu entitled "Optimizing Strategic Planning With Long-term Sequential Decision Making Under Uncertainty: A Decomposition Approach." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Industrial Engineering.

Xueping Li and Anahita Khojandi, Major Professors

We have read this thesis
and recommend its acceptance:

Xueping Li

Anahita Khojandi

Olufemi Omitaomu

Shuai Li

Accepted for the Council:

Dixie Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Optimizing Strategic Planning With Long-term Sequential Decision Making Under Uncertainty: A Decomposition Approach

A Dissertation Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Zeyu Liu

August 2022

for my mother & father,
and all members in my family

Acknowledgements

I would like to thank Dr. Xueping Li and Dr. Anahita Khojandi, for your support, your guidance, and your unwavering confidence in me, without whom I could not have become the same person as I am.

I would like to thank Guirong Gong and Hao Liu, my parents, for everything.

I would like to thank Sarita Rattanakunuprakarn, Ziwei Liu, and Xudong Wang, for your friendship and all the happy memories of us.

I would like to thank Dr. Yang Yang, for being the light in my darkest hours.

This work was supported in part by the Graduate Advancement, Training and Education (GATE) Program between the Oak Ridge National Laboratory and the University of Tennessee, Knoxville.

per aspera ad astra

Abstract

The operations research literature has seen decision-making methods at both strategic and operational levels, where high-level strategic plans are first devised, followed by long-term policies that guide future day-to-day operations under uncertainties. Current literature studies such problems on a case-by-case basis, without a unified approach. In this study, we investigate the joint optimization of strategic and operational decisions from a methodological perspective, by proposing a generic two-stage long-term strategic stochastic decision-making (LSSD) framework, in which the first stage models strategic decisions with linear programming (LP), and the second stage models operational decisions with Markov decision processes (MDP). The joint optimization model is formulated as a nonlinear programming (NLP) model, which is then reduced to an integer model through discretization.

As expected, the LSSD framework is computationally expensive. Thus, we develop a novel solution algorithm for MDP, which exploit the Benders decomposition with the "divide-and-conquer" strategy. We further prove mathematical properties to show that the proposed multi-cut L-shaped (MCLD) algorithm is an exact algorithm for MDP. We extend the MCLD algorithm to solve the LSSD framework by developing a two-step backward decomposition (TSBD) method. To evaluate algorithm performances, we adopt four benchmarking problems from the literature. Numerical experiments show that the MCLD algorithm and the TSBD method outperform conventional benchmarks by up to over 90% and 80% in algorithm runtime, respectively.

The practicality of the LSSD framework is further validated on a real-world critical infrastructure systems (CISs) defense problem. In the past decades, "attacks" on CIS facilities from deliberate attempts or natural disasters have caused disastrous consequences all over the globe. In this study, we strategically design CIS interconnections and allocate defense resources, to protect the CIS network from sequential, stochastic attacks. The LSSD framework is utilized to model the problem as an NLP model with an alternate integer formulation. We estimate model parameters using real-world CIS data collected from a middle-sized city in the U.S. Previously established algorithms are used to solve the problem with over 45% improvements in algorithm runtime. Sensitivity analyses are conducted to investigate model behaviors and provide insights to practitioners.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The combined optimization of strategic planning and long-term sequential decision making has been studied in many distinct application areas in operations research. For example, in a power generator placement problem (Kizito et al. 2021), the decision maker plans the strategic locations of the power generators in the present, by considering the minimum cost in future operations that satisfies customer demands during a potential power outage lasting for days. In another study that models infrastructure flexibility (Torres-Rincón et al. 2021), the decision maker first considers the optimal infrastructure network design that requires strategic investment, and then optimizes flows in the network sequentially for multiple time periods in the future. A problem with a similar structure is also studied in farmland irrigation management (Li and Hu 2020), where the decision maker decides seed types and plant densities before the farming season begins, and then sequentially chooses the optimal timing of irrigation and the quantity of water during the farming season.

Although scattered among different application areas, these studies feature a common decision making paradigm, where the decision maker optimizes for a strategic decision that should be implemented in the present with substantial investments, as well as a future operational policy that supports the functionality of the system in the long term. The strategic decision is usually associated with relatively heavy investment, and imposes constraints and impacts on the long-term operational policy. In this case, to search for an optimal solution, the decision maker must consider the combined optimization of both the strategic decision and the future operational policy, by evaluating the cost-benefit between different long-term operational policies in order to choose the strategic decision in the present. The joint decision-making process is illustrated in Figure 1.1.

A simple example would be a machine purchase & maintenance problem (Puterman 2014). Suppose that a factory wishes to purchase machines to satisfy production needs. It is apparent that the future production level heavily depends on the current purchase decision. But to reach a long-term profitable state, the factory also has to consider regular maintenance strategies for the machines, so that the production level will not be affected by machine failure. Thus, although purchasing cheaper machines seems to be a better choice myopically, the low reliability and constant needs for maintenance of the machines could instead cost more in the long-term, potentially making purchasing expensive machines a more reasonable choice.

Similar to purchasing machines, strategic decisions often require substantial investments, such as constructing facilities, designing logistics or utility networks, investing capital, etc. The strategic decisions are typically only made once at the beginning of the entire decision-making horizon and are assumed to be fixed afterward due to the high cost of modification. In practice, the strategic decisions can be modeled using mathematical programming methods, e.g., linear programming (LP), which has been extensively applied in the literature to a variety of applications in the past decades (Bertsimas and Tsitsiklis 1997).

The long-term operational policy can be modeled by sequential decision making models, which have long been a centerpiece in operations research. The popularity of sequential decision making is indispensable to its successful applications in many areas, such as manufacturing (Kazemi Zanjani et al. 2010), finance (Mulvey and Shetty 2004), transportation (Delgado et al. 2019), healthcare (Ayer et al. 2012), risk management (Ruszczyński 2010; Fan and Ruszczyński 2018), and artificial intelligence (Mnih et al. 2015; Silver et al. 2017). Sequential decision making models become especially useful for practical applications when uncertainties in constantly evolving environments are incorporated. Such models often allow decision makers to dynamically prescribe optimal decisions facing different

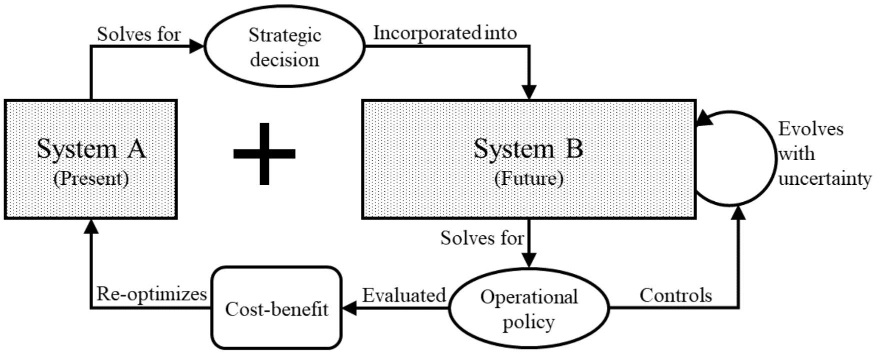


Figure 1.1: A high-level demonstration of the proposed two-stage framework.

situations. As a result, sequential decision making under uncertainty features a combination of determinism and stochasticism, where the model recommends a deterministic decision (whether based on some probability distribution or not) under the current situation, and at the same time considers the impact on all possible scenarios in an uncertain future.

Among many, stochastic programming has become one of the most adopted approaches to model sequential decisions under uncertainty. In stochastic programming, at each decision making epoch, a mathematical program is constructed for every scenario, resulting in a tree-structured multistage model detailing the sample paths of all realizations of possible scenarios (Birge and Louveaux 2011). A primary advantage of multistage stochastic programming lies in its adroitness in modeling complex decisions with intricate system dynamics through a variety of variables and constraints. However, a tree-structured scenario realization results in an exponentially expanding model, scaled up with the number of decision epochs, limiting the potential of practical implementations to large-scale problems. In fact, the applications of multistage stochastic programming in many recent studies are restricted to less than 10 stages due to the expensive computation of large models (Delgado et al. 2019; Kıbış et al. 2020; Kuhn 2008; Yin and Büyüktahtakın 2021).

Besides stochastic programming, the Markov decision process (MDP) is another commonly adopted approach to sequential decision making. At its core, MDP utilizes a Markov process to model the dynamic transitions between system states, and makes optimal decisions based on current system states with respect to a decision rule to ensure a maximized aggregated reward in the entire process (Puterman 2014). Unlike stochastic programming, MDP is widely used to model long-term decision making problems due to the guaranteed existence of a stationary policy, i.e., a decision making rule independent of the decision epochs (Howard 1960). Thus, MDP is commonly solved by dynamic programming algorithms (Bellman 1957), saving the expensive computational requirement of going through tree-structured models. In particular, a previous study has shown the advantage of dynamic programming over multistage stochastic programming, especially in longer planning horizons (Archibald et al. 1999).

In the literature, several studies have pioneered the hierarchical modeling that utilizes LP to optimize strategic decisions, and MDP for operational decisions. For example, in a critical infrastructure protection problem, network design strategies are extended with resource allocation decisions to maximize intrusion detection (Jones et al. 2006). The "upper level" optimization uses LP to model resource allocation and the "lower level" optimization uses MDP to model intrusions to a facility under a stochastic environment. A similar framework has been proposed for a production process to optimize revenue management policies (Cooper and Mello 2007), in which the sequential decision making horizon is decomposed into two portions, one using LP to coordinate commodity production and another using MDP to satisfy arriving customer demands. The combined optimization of LP and MDP is also applied to optimize a photovoltaic (PV) system with energy storage (Keerthisinghe et al. 2014). The proposed optimization framework solves stochastic linear programs for longer horizons, and solves MDP for shorter time periods within the longer horizon.

Although the above studies have shown the practicality of combining LP with MDP in modeling complex systems, the method still has limitations. By modeling a problem as an MDP, the structure of the problem must fit a specific paradigm, in which the essential elements of MDP, such as states, actions, transition probabilities, and rewards, have to be explicitly defined. This sometimes makes modeling with MDP difficult, especially for complex systems. Although constraints are allowed in the LP formulation of MDP (Manne 1960; d'Epenoux 1960; Oliver 1960), the resulting models, such as constrained MDP (CMDP), often focus on a particular type of constraints that impose upper bounds on the policy (Derman and Klein 1965). Thus, compared with stochastic programming, constraints that model complex system dynamics and sophisticated relationships between variables are difficult to incorporate into MDP. Moreover, large-scale MDP problems are typically hard to solve because of the curse of dimensionality, where the scales of the models grow so large that traditional algorithms become intractable (de Farias and Van Roy 2003). The negative impacts of model scale are more significant on CMDP, for which the mainstream exact solution method remains to be the LP formulation (Altman 1999).

In fact, studies in the literature have pointed out solution algorithms as the main difficulty in implementing the decision-making framework that combines LP with MDP (Jones et al. 2006). As a result, current studies mostly choose to optimize the LP and MDP models separately, without considering the global optimum of the two joint systems (Cooper and Mello 2007; Keerthisinghe et al. 2014). Indeed, current solution methods for MDP show disadvantages when facing another linear system. State-of-the-art exact solution algorithms such as value iteration (VI) or policy iteration (PI) do not guarantee full support on linear constraints. Variants in the literature often solve for near-optimal solutions using approximate dynamic programming (ADP) or reinforcement learning (RL) techniques (Achiam et al. 2017; Vieillard et al. 2019). The gaps in the literature not only call for the generic formulation and analytical results of the joint decision-making approach, but also exact solution algorithms that solve the joint model to the true global optimum.

In this study, we aim to fill in the gaps in the literature by proposing a generic mathematical formulation that optimizes strategic decisions with sequential operational decisions for a stochastic system in the long term. Specifically, we formulate the long-term strategic stochastic decision-making (LSSD) framework by combining an LP model with MDP. The LP model makes strategic decisions that have immediate impacts on the MDP, which optimizes sequential, operational decisions for the stochastic system. In addition, we also consider two extensions of the LSSD framework based on CMDP, where additional linear constraints are added to the model to regulate the operational policy.

Facing the computational difficulties in solving the LSSD framework, we develop novel exact algorithms that find the global optimum of the LSSD framework. We first investigate a more efficient way of solving MDP exactly in its LP formulation, by exploiting the Benders decomposition (Benders 1962). The novel solution method solves MDP much more efficiently than brute-forcing the LP formulation directly. More importantly, unlike conventional solution methods, the decomposition algorithm allows us to incorporate other linear systems and optimize them in a joint way. Thus, we extend the novel algorithm for MDP to the LSSD framework and develop decomposition methods to solve the framework and its extensions.

We further conduct computational analyses to evaluate the algorithm performances of the proposed methods, using four benchmarking problems from the literature.

To further validate the LSSD framework, we apply the framework to a real-world problem that protects interconnected critical infrastructure systems (CIS) from sequential attacks, where the defender only receives stochastic information on the attacker's intention. The CIS protection problem is modeled using the LSSD framework, where strategic decisions are made regarding defense resource allocation and CIS network design, and operational decisions are made regarding the optimal defense strategies in response to the attacks. We collect CIS data from a middle-sized city in the U.S., and conduct thorough estimations of model parameters, especially the consequences of successful attacks, using real data as well as data from the literature. Previously developed algorithms are implemented and validated using the CIS protection problem. We have also conducted additional numerical experiments to draw insights from the results of the LSSD framework for government agencies to coordinate resources in response to the attacks on CIS networks.

We summarize the main contributions of this study to the current literature from both the methodological perspective and the practical perspective. The methodological contributions are as follows.

1. We propose a generic modeling framework, LSSD, that jointly optimizes strategic planning with long-term sequential decision making under uncertainty;

2. We formulate the LSSD framework mathematically by hierarchically combining an LP model with MDP;

3. We extend the LSSD framework by introducing additional linear constraints;

4. We transform LSSD into an alternative formulation to reduce nonlinearity, using discretization and integer variables;

5. We propose a novel generic decomposition method that solves the LP formulation of MDP efficiently;

6. We prove mathematical properties of the decomposition method and show that it is an exact method that solves MDP; and

7. We extend the algorithm to solve the LSSD framework, greatly reducing the computational complexity.

The practical contributions of this study are summarized as follows.

1. We include and extend four benchmarking problems from the literature to evaluate algorithm performances;

2. We conduct computational analysis and show that our proposed algorithms solve MDP and the LSSD framework significantly faster than the benchmarking methods;

3. We apply the proposed methodology to a real-world CIS defense problem under stochastic sequential attacks, featuring interconnectivity between CIS facilities, as well as long-term dynamic defense strategies;

4. We conduct a case study by collecting real-world data from a middle-sized city in the U.S., and performing a thorough estimation on model parameters;

5. We design and conduct experiments to solve the CIS protection problem and show the advantages of our proposed methods and algorithms; and

6. We conduct sensitivity analyses on five different model parameters. Insights are provided to practitioners through thoroughly investigating model behaviors.

This dissertation is structured into six chapters. In Chapter 1, we lay the foundation by introducing the research problem and discussing relevant studies in the literature. In Chapter 2, we formulate the mathematical model of the LSSD framework and consider the extensions based on CMDP. We further analyze the framework using a decomposition method and propose alternate formulations that reduce the nonlinearity in the models. In Chapter 3, we first develop a novel decomposition algorithm for the MDP, and then extend the algorithm to solve the LSSD framework as well as its extensions. In Chapter 4, we conduct computational analysis using benchmarking problems from the literature to evaluate the algorithms developed in the previous chapter. We also design experiments to analyze the algorithm behavior in detail. In Chapter 5, we apply the LSSD framework to the CIS protection problem. We formulate the mathematical framework, estimate model parameters, optimize with developed algorithms, and conduct experiments to analyze model results. In Chapter 6, we draw conclusions and provide insights.

Chapter 2

Model Formulation

In this chapter, we formulate the mathematical model of the LSSD framework and present two extensions based on CMDP. We further analyze the model using a decomposition method and propose an alternate formulation that reduces nonlinearity in the model. Alternate formulations are also applied to the extensions. We begin this chapter by introducing the notation of MDP.

2.1 MDP

We define an MDP as a five-element tuple (S, A, T, R, γ), where S denotes the set of system states, A denotes the set of available actions, $T : S \times S \times A \to \mathbb{R}_+$ denotes the transition probability, $R : S \to A$ denotes the reward function and γ denotes the discount factor (Puterman 2014). In addition, we let $\boldsymbol{\alpha} \in \mathbb{R}_+^{|S|}$ be the initial state distribution, i.e., $0 \preceq \boldsymbol{\alpha} \preceq 1, \boldsymbol{\alpha}^T \cdot \mathbf{1} = 1$. Note that we use bold letters to represent the vector form of variables and parameters, and we use letters with subscripts, e.g. α_s, $\forall s \in S$, to denote an element in the vector. In this study, we let the action set A be independent of the state set S, which includes the situation where some actions are not available under specific states, since one can easily set the state transition probability to 0 for unavailable actions.

Specifically, we focus on infinite horizon MDP with discount $(0 < \gamma < 1)$ that makes sequential decisions at the decision epochs $t = 0, 1, 2, ..., \infty$. The decisions are made according to a policy $\pi : S \rightarrow A$, which takes an action $a_t \in A$ based on the current system state $s_t \in S$. After an action is taken, the system obtains a reward $R(s_t, a_t)$. Then, the system transitions to a new state s_{t+1} at the next decision epoch with the transition probability $T(s_{t+1}|s_t, a_t)$. The objective of the MDP is to maximize the accumulated reward over the entire decision making horizon. As such, the value function $V_t : S \rightarrow \mathbb{R}$ of the MDP can be defined as the accumulated reward-to-go starting from t,

$$V_t(s) = R(s_t, a_t) + \gamma \sum_{s' \in S} T(s'|s_t, a_t) V^*_{t+1}(s'), \quad \forall\ s \in S, \tag{2.1.1}$$

where $V^*_t(s)$ denotes the Bellman optimality equation (Bellman 1957),

$$V^*_t(s) = \max_{a \in A} \left\{ R(s_t, a_t) + \gamma \sum_{s' \in S} T(s'|s_t, a_t) V^*_{t+1}(s') \right\}, \quad \forall\ s \in S. \tag{2.1.2}$$

It has been proved that the optimal solution, i.e., a policy π^* that solves $V^*_0(s)$, $\forall\ s \in S$, always exists for the infinite horizon MDP (Howard 1960).

Since the Bellman optimality equation uses dynamic programming in essence, there exists an LP equivalent to the infinite horizon MDP (Manne 1960; d'Epenoux 1960; Oliver 1960). Let $\boldsymbol{v} \in \mathbb{R}^{|S|}$ denote the value of each state and V^* the value of the MDP, the linear program can be written as follows:

$$V^* := \min \quad \sum_{s \in S} \alpha_s v_s \tag{2.1.3}$$

$$\text{s.t.} \quad v_s - \gamma \sum_{s' \in S} T(s'|s, a) v_{s'} \geq R(s, a), \quad \forall\ s \in S, a \in A; \tag{2.1.4}$$

$$v_s \text{ unrestricted}, \quad \forall\ s \in S. \tag{2.1.5}$$

In the following, we refer to this linear program as the LP formulation of MDP. The LP formulation constructs a constraint for each state-action pair (s, a), resulting in a total of $|S| \cdot |A|$ constraints. Thus, the size of the LP formulation scales nonlinearly with the size of the states and actions, making it difficult to optimize for large-scale implementations.

2.2 Formulation of The LSSD Framework

We formulate the LSSD framework as a two-stage model. As discussed in the previous chapter, we model the strategic decisions in the first stage with LP, and the operational decisions in the second stage with MDP. In the first stage, we consider a generic LP model to promote applications to different practical problems. Let $\boldsymbol{x} \in \mathbb{R}^n_+$ be the decision making variable representing the strategic decisions with dimension n. The generic LP model can be written as (Bertsimas and Tsitsiklis 1997)

$$\max \quad \sum_{i=1}^{n} c_i x_i \tag{2.2.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\, j = 1, \dots, m; \tag{2.2.2}$$

$$x_i \geq 0 \quad \forall\, i = 1, \dots, n, \tag{2.2.3}$$

where $w_{i,j}$ denotes an element in the coefficient matrix $W_{m \times n}$ and $\boldsymbol{b}$ denotes a vector of the right-hand-side constraints.

The second stage makes sequential decisions through an MDP. Note that in the LP formulation of MDP, i.e., Equation (2.1.3) – (2.1.5), the objective is to minimize the state values, even though the objective of MDP is to maximize the accumulated reward. The LP formulation of MDP functions well on its own with the minimizing objective. However, when combining the LP formulation with another linear system, the objective causes conflict between the two systems, especially when the other system changes the parameters of the MDP. As a result, a minimizing objective makes the model choose the set of parameters with the worst objective value. Thus, instead of the LP formulation of MDP, we consider the following dual problem of the LP formulation of MDP (Puterman 2014).

$$\max \quad \sum_{s \in S} \sum_{a \in A} R(s,a) y_{s,a} \tag{2.2.4}$$

$$\text{s.t.} \quad \sum_{a \in A} y_{s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} T(s|s',a) y_{s',a} = \alpha_s \quad \forall\, s \in S; \tag{2.2.5}$$

$$y_{s,a} \geq 0 \quad \forall\ s \in S, a \in A, \tag{2.2.6}$$

where the decision-making variable $y_{s,a}$ represents an occupation measure, suggesting the number of times that action a is taken under state s (Dolgov and Durfee 2005).

In this study, we make an important assumption about the relationship between the strategic decisions $\boldsymbol{x}$ and the MDP model. Specifically, we assume that the state set S, the action set A, and the discount factor γ are not affected by the values of $\boldsymbol{x}$.

Assumption 2.1. *The state set S, the action set A, and the discount factor γ in the second stage MDP remain fixed for all possible values of $\boldsymbol{x}$.*

Assumption 2.1 are widely adopted by the literature regarding MDP parameter ambiguity (Satia and Lave Jr 1973; Mannor et al. 2007; Bäuerle and Rieder 2019; Steimle et al. 2021b). Even though in practice, strategic decisions may lead to different a state or action set, it is always possible to model the MDP in such a way that S and A contain all possible states and actions no matter the value of $\boldsymbol{x}$. States that cannot be visited or actions that cannot be taken are then modeled by setting the state transitions to 0, or the rewards to negative infinity.

As such, the strategic decision $\boldsymbol{x}$ only affects the transition probability $T(s'|s,a)$ and the reward $R(s,a)$. Since the values of $T(s'|s,a)$ and $R(s,a)$ change, we introduce variables $\tau_{s',s,a} \in [0,1]$, $\forall\ s', s \in S, a \in A$ and $r_{s,a} \in \mathbb{R}$, $\forall\ s \in S, a \in A$ to represent the values of $T(s'|s,a)$ and $R(s,a)$, respectively, under the influence of $\boldsymbol{x}$. Further, the effect of $\boldsymbol{x}$ on $\tau_{s',s,a}$ and $r_{s,a}$ are considered in a generic way. We introduce functions $G : \mathbb{R}_+^n \times [0,1]^{S\times S\times A} \to \mathbb{R}$ and $H : \mathbb{R}_+^n \times \mathbb{R}^{S\times A} \to \mathbb{R}$ to model the connections between the first and the second stage. Specifically, the values of $\tau_{s',s,a}$ and $r_{s,a}$ are subject to the following constraints.

$$\begin{aligned}
&G(\boldsymbol{x}, \boldsymbol{\tau}) = 0; \\
&H(\boldsymbol{x}, \boldsymbol{r}) = 0; \\
&\sum_{s' \in S} \tau_{s',s,a} = 1 \quad \forall\ s \in S, a \in A.
\end{aligned}$$

Using the above notation, the mathematical model of the two-stage LSSD framework is formulated as the following nonlinear program (NLP).

$$
\begin{aligned}
\text{NLP} := \max \quad & \sum_{i=1}^{n} c_i x_i + \sum_{s \in S} \sum_{a \in A} r_{s,a} y_{s,a} && (2.2.7)\\
\text{s.t.} \quad & \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\, j = 1, \ldots, m; && (2.2.8)\\
& G(\boldsymbol{x}, \boldsymbol{\tau}) = 0; && (2.2.9)\\
& H(\boldsymbol{x}, \boldsymbol{r}) = 0; && (2.2.10)\\
& \sum_{s' \in S} \tau_{s',s,a} = 1 \quad \forall\, s \in S, a \in A; && (2.2.11)\\
& \sum_{a \in A} y_{s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} \tau_{s,s',a} y_{s',a} = \alpha_s \quad \forall\, s \in S; && (2.2.12)\\
& \boldsymbol{x}, \boldsymbol{y} \succeq \boldsymbol{0}, \boldsymbol{\tau}, \boldsymbol{r} \text{ unrestricted.} && (2.2.13)
\end{aligned}
$$

In the model, nonlinearity arises when we combine an LP model with MDP. Specifically, in the first stage, the strategic decision $\boldsymbol{x}$ leads to different choices of $\tau_{s',s,a}$ and $r_{s,a}$. Then, in order to calculate the optimal policy, $\tau_{s',s,a}$ and $r_{s,a}$ are multiplied with the second-stage decision-making variable $y_{s,a}$ in the objective as well as Constraint (2.2.12), leading to nonlinear terms in the model.

2.3 Extension to CMDP

The practical significance of CMDP arises from the areas where general MDPs tend to be inadequate. Especially, when an MDP is optimized subject to multiple objectives (Armony and Ward 2010; Boussard and Miura 2011), or when certain resource limitations are present (Bhandari et al. 2008; Chen et al. 2018), additional constraints are required to eliminate infeasible policies. Due to the uniqueness of applications, CMDP has seen different formulations in the literature (Altman 1999; Heyman and Sobel 2004), all of which produce the same optimal policy in the essence. For consistency, in this study, we adopt the formulation closely related to the dual problem of an unconstrained MDP (Heyman and Sobel 2004; Dolgov and Durfee 2005).

Let $d_i : S \times A \rightarrow \mathbb{R}$, $i \in \mathcal{D}$, be the cost functions of taking an action under a state, where $\mathcal{D}$ is the set of constraint indices, representing $|\mathcal{D}|$ types of costs, each associated with a different upper bounds ("budgets") $\bar{D}_i \in \mathbb{R}$, $i \in \mathcal{D}$. Using the dual formulation of unconstrained MDP, CMDP can be represent as follows.

$$\max \quad \sum_{s\in S}\sum_{a\in A} R(s,a) y_{s,a} \tag{2.3.1}$$

$$\text{s.t.} \quad \sum_{a\in A} y_{s,a} - \gamma \sum_{s'\in S}\sum_{a\in A} T(s|s',a) y_{s',a} = \alpha(s), \quad \forall\, s \in S; \tag{2.3.2}$$

$$\sum_{s\in S}\sum_{a\in A} d_i(s,a) y_{s,a} \le \bar{D}_i, \quad \forall\, i \in \mathcal{D}; \tag{2.3.3}$$

$$y_{s,a} \ge 0, \quad \forall\, s \in S, a \in A. \tag{2.3.4}$$

Similar to the dual formulation, $y_{s,a}$ is the decision-making variable, representing an occupation measure, i.e., the number of times the action a is taken under state s (Dolgov and Durfee 2005). Thus, the additional constraint (2.3.3) can be interpreted as a resource limitation applied to the aggregated actions taken under all states.

The integration of CMDP with another linear system becomes an easy extension based on the formulation (2.2.7) – (2.2.13). The following model shows the formulation of combining a linear system of strategic decision-making with CMDP (NLP-C).

$$\text{NLP-C} := \max \quad \sum_{i=1}^{n} c_i x_i + \sum_{s\in S}\sum_{a\in A} r_{s,a} y_{s,a} \tag{2.3.5}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\, j = 1, \ldots, m; \tag{2.3.6}$$

$$G(\boldsymbol{x}, \boldsymbol{\tau}) = 0; \tag{2.3.7}$$

$$H(\boldsymbol{x}, \boldsymbol{r}) = 0; \tag{2.3.8}$$

$$\sum_{s'\in S} \tau_{s',s,a} = 1 \quad \forall\, s \in S, a \in A; \tag{2.3.9}$$

$$\sum_{a \in A} y_{s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} \tau_{s,s',a} y_{s',a} = \alpha_s \quad \forall\, s \in S; \tag{2.3.10}$$

$$\sum_{s \in S} \sum_{a \in A} d_i(s,a) y_{s,a} \leq \bar{D}_i, \quad \forall\, i \in \mathcal{D}; \tag{2.3.11}$$

$$\boldsymbol{x}, \boldsymbol{y} \succeq \mathbf{0}, \boldsymbol{\tau}, \boldsymbol{r} \text{ unrestricted.} \tag{2.3.12}$$

The additional constraint, i.e., Constraint (2.3.11), imposes a heavier computational cost on top of the original nonlinear system.

2.4 Extension to CMDP With Variable Budgets

Often, when conducting strategic planning, decision makers are not only concerned with future revenue, but also with the present investment of implementing the decisions (Blumentritt 2006). The "perfect" future operational policy becomes useless if the current budget does not permit it. A practical example would be the bidding for contracts, where tenders submit proposals or quotations in response to solicitations from contracting authority (Samuelson 1986). In the context of CMDP, the budget of each proposal can be seen as a possible value for $\bar{D}_i$. Since the decision maker needs to find the optimal "bid" from multiple CMDP "tenders" in the second stage, we consider $\bar{D}_i$ as another decision-making variable in the first stage.

Specifically, we let $\bar{D}_i$ be the maximum budget that the decision maker is willing to pay, and use $D_i \in \mathbb{R}$, $i \in \mathcal{D}$ as an additional variable that connects the two stages of the model. In addition, we introduce a weight coefficient $\eta \in [0,1]$, representing the importance of the budget to the decision maker. The extension to CMDP with variable budgets (NLP-VB) can be formulated as the following program.

$$\text{NLP-VB} := \max \quad \sum_{i=1}^{n} c_i x_i - \eta \sum_{i \in \mathcal{D}} D_i + \sum_{s \in S} \sum_{a \in A} r_{s,a} y_{s,a} \tag{2.4.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\, j = 1, \ldots, m; \tag{2.4.2}$$

$$G(\boldsymbol{x}, \boldsymbol{\tau}) = 0; \tag{2.4.3}$$

$$H(\boldsymbol{x}, \boldsymbol{r}) = 0; \tag{2.4.4}$$

$$\sum_{s' \in S} \tau_{s',s,a} = 1 \quad \forall\ s \in S, a \in A; \tag{2.4.5}$$

$$\sum_{a \in A} y_{s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} \tau_{s,s',a} y_{s',a} = \alpha_s \quad \forall\ s \in S; \tag{2.4.6}$$

$$\sum_{s \in S} \sum_{a \in A} d_i(s,a) y_{s,a} \leq D_i \quad \forall\ i \in \mathcal{D}; \tag{2.4.7}$$

$$D_i \leq \bar{D}_i \quad \forall\ i \in \mathcal{D}; \tag{2.4.8}$$

$$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{D} \succeq \boldsymbol{0}, \boldsymbol{z} \in \{0,1\}^K, \boldsymbol{\tau}, \boldsymbol{r} \text{ unrestricted.} \tag{2.4.9}$$

In the model, the occupation measure $y_{s,a}$ in the second stage is constrained to be less than D_i, and the objective in the first stage seeks to find the MDP with the minimum D_i. When $\eta = 1$, the decision maker evaluates the budget to the full extent. When $\eta = 0$, the model is equivalent to the formulation (2.3.5) – (2.3.12) without the variable budgets.

2.5 Decomposing The LSSD Framework

In this section, we analyze the previously proposed models, in particular, the formulation (2.2.7) – (2.2.13), as it is the foundation of the two extensions. Specifically, we apply the generalized Benders decomposition technique to the model formulation (Geoffrion 1972), with the intent to reduce the nonlinearity in the objective as well as constraints.

The generalized Benders decomposition divides a mathematical program into two parts, a master problem (MP) and a subproblem (SP). This decomposition method is widely adopted in the literature when the model contains nonlinear constraints or objectives, rendering the regular Benders decomposition non-applicable. By applying the generalized Benders decomposition, when the solution to the MP is fixed, the SP is transformed into a linear system, allowing it to be solved using conventional algorithms, reducing the nonlinearity in the model (Geoffrion 1972).

In our model, the MP and SP corresponds to the first stage (LP) and the second stage (MDP) formulation. The MP contains the strategic decision $\boldsymbol{x}$ as well as the connecting variables $\boldsymbol{\tau}$ and $\boldsymbol{r}$, with an additional variable θ representing the value of the subproblem.

$$\begin{aligned}
\max \quad & \sum_{i=1}^{n} c_i x_i + \theta && (2.5.1)\\
\text{s.t.} \quad & \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\ j = 1, \ldots, m; && (2.5.2)\\
& G(\boldsymbol{x}, \boldsymbol{\tau}) = 0; && (2.5.3)\\
& H(\boldsymbol{x}, \boldsymbol{r}) = 0; && (2.5.4)\\
& \sum_{s' \in S} \tau_{s',s,a} = 1 \quad \forall\ s \in S, a \in A; && (2.5.5)\\
& \boldsymbol{x} \succeq \mathbf{0}, \boldsymbol{\tau}, \boldsymbol{r}, \theta \text{ unrestricted.} && (2.5.6)
\end{aligned}$$

Let $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$ be the solutions obtained from the MP. The SP uses the solution to formulate the MDP:

$$\begin{aligned}
\theta := \max \quad & \sum_{s \in S} \sum_{a \in A} \bar{r}_{s,a} y_{s,a} && (2.5.7)\\
\text{s.t.} \quad & \sum_{a \in A} y_{s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} \bar{\tau}_{s,s',a} y_{s',a} = \alpha_s \quad \forall\ s \in S; && (2.5.8)\\
& \boldsymbol{y} \succeq \mathbf{0}. && (2.5.9)
\end{aligned}$$

As such, the nonlinearity in the original formulation is alleviated, as the connecting variables $\boldsymbol{\tau}$ and $\boldsymbol{r}$ are converted into coefficients, instead of variables.

In order to obtain the optimal solution to the decomposed model, Additional constraints are derived from the SP and added back to the MP (Geoffrion 1972). The MP is then solved to produce new solutions of $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$, from which a new SP is formulated. The process continues as an iterative algorithm until convergence conditions are met. Typically, two types of constraints, or cuts, are derived, i.e., feasibility cuts and optimality cuts. The feasibility cuts ensure that the solutions produced by the first stage are feasible for the second stage.

In our model, since the second stage is an MDP, it is always feasible no matter the first-stage solutions. Thus the following proposition of complete recourse holds (Birge and Louveaux 2011).

Proposition 2.1. *The SP (2.5.7) – (2.5.9) has complete recourse, i.e., ∀ $\bar{\boldsymbol{\tau}}, \bar{\boldsymbol{r}}$, there always exists a feasible solution $\bar{\boldsymbol{y}}$.*

Proof. The proposition holds since model (2.5.7) – (2.5.9) is an infinite-horizon, discount MDP. It has been proved that optimal policy for an infinite-horizon, discount MDP always exists (Howard 1960). Thus, there always exists a feasible solution to the model (2.5.7) – (2.5.9). □

Proposition 2.1 suggests that no feasibility cuts are required to reach the optimal solution, and only optimality cuts need to be derived. To do so, we apply Lagrangian relaxation to the SP. Let v_s be the Lagrangian multiplier corresponding to the Constraint (2.5.8). The SP is relaxed to an unconstrained optimization problem.

$$\sup_{\boldsymbol{y}\succeq\boldsymbol{0}} \Big\{ \sum_{s\in S}\sum_{a\in A} r_{s,a} y_{s,a} + \sum_{s\in S} v_s \cdot (\alpha_s - \sum_{a\in A} y_{s,a} - \gamma \sum_{a\in A}\sum_{s'\in S} \tau_{s,s',a} y_{s',a}) \Big\}. \tag{2.5.10}$$

The value v_s can be obtained from the SP easily using duality with fixed $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$. With a simple transformation, the optimality cuts can be formulation as follows.

$$\theta \ \leq \ \sum_{s\in S}\sum_{a\in A} \sup_{\boldsymbol{y}\succeq\boldsymbol{0}} \Big\{ (r_{s,a} - v_s + \gamma \sum_{s'\in S} \tau_{s',s,a} v_{s'}) y_{s,a} \Big\} + \sum_{s\in S} \alpha_s v_s, \tag{2.5.11}$$

Then, the MP to be solved iteratively takes the following form.

$$\max \quad \sum_{i=1}^{n} c_i x_i + \theta \tag{2.5.12}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\ j = 1, \ldots, m; \tag{2.5.13}$$

$$G(\boldsymbol{x}, \boldsymbol{\tau}) = 0; \tag{2.5.14}$$

$$H(\boldsymbol{x}, \boldsymbol{r}) = 0; \tag{2.5.15}$$

$$\sum_{s' \in S} \tau_{s',s,a} = 1 \quad \forall\, s \in S, a \in A; \tag{2.5.16}$$

$$\theta \;\leq\; \sum_{s \in S} \sum_{a \in A} \sup_{\boldsymbol{y} \succeq \boldsymbol{0}} \Big\{ (r_{s,a} - v_s + \gamma \sum_{s' \in S} \tau_{s',s,a} v^{\ell}_{s'}) y_{s,a} \Big\} + \sum_{s \in S} \alpha_s v^{\ell}_s \quad \forall\, \ell = 1, \ldots, L \tag{2.5.17}$$

$$\boldsymbol{x} \succeq \boldsymbol{0}, \boldsymbol{\tau}, \boldsymbol{r}, \theta \text{ unrestricted,} \tag{2.5.18}$$

where $\ell = 1, 2, \ldots, L$ denotes the number of iterations. Note that the optimality cut (2.5.11) is formulated in an intuitive way. The term $\sum_{s \in S} \alpha_s v_s$ matches the objective of the primal LP formulation of MDP. The term $r_{s,a} - v_s + \gamma \sum_{s' \in S} \tau_{s',s,a} v_{s'}$ matches the constraints of the primal LP formulation of MDP. Such duality occurs in the cut since it is derived through the Lagrangian method.

However, generalized Benders decomposition does not fully remove nonlinearity from the model. The supremum in the optimality cut (2.5.11) cannot be evaluated in a closed form, since the sign of the term $r_{s,a} - v_s + \gamma \sum_{s' \in S} \tau_{s',s,a} v_{s'}$ remains inconclusive. Considering that the supremum is taken with respect to $\boldsymbol{y} \succeq \boldsymbol{0}$, we can characterize the supremum in the following way.

$$\sup_{\boldsymbol{y} \succeq \boldsymbol{0}} \Big\{ (r_{s,a} - v_s + \gamma \sum_{s' \in S} \tau_{s',s,a} v_{s'}) y_{s,a} \Big\} = \begin{cases} \infty & \text{if } r_{s,a} - v_s + \gamma \sum_{s' \in S} \tau_{s',s,a} v_{s'} > 0; \\ 0 & \text{otherwise.} \end{cases} \tag{2.5.19}$$

When $r_{s,a} - v_s + \gamma \sum_{s' \in S} \tau_{s',s,a} v_{s'} > 0$, the optimality cut does not bind in the MP, since the cut is equivalent to $\theta \leq \infty$. Interestingly, if $\bar{r}_{s,a} - v_s + \gamma \sum_{s' \in S} \bar{\tau}_{s',s,a} v_{s'} > 0$ holds for some $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$, it also suggests that $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$ violate the feasibility of the constraints in a primal formulation with state value v_s. Thus, $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$ are a pair of MDP parameters that have never been evaluated by the optimality cuts added in the previous iterations. As a result, such a pair of parameters would provide the MP with a larger objective, with fewer binding constraints. Note that by modeling Equation (2.5.19) using integer programming (IP) techniques such as the big-M method, it is possible to approximate the supremum with integer variables. However, doing so trades one form of nonlinearity with another, not necessarily helping reduce the computational complexity of the model.

Nonetheless, Equation (2.5.19) still provides us with intuitive insights regarding how generalized Benders decomposition behaves when applied to the two-stage LSSD framework. Specifically, the MP would prefer to produce a pair of parameters $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$ that have never been evaluated in previous iterations, which provides a better objective value. Then, the SP evaluates such a pair of parameters and produce an optimality cut (2.5.11) that binds the current pair of $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$ with the value $\sum_{s \in S} \alpha_s v_s$, which is also the objective of the primal LP formulation of MDP. In a new iteration, the MP would search for another pair of $\bar{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{r}}$ representing a new MDP. Finally, when all possible sets of MDP parameters generated by the first stage are evaluated, one after another, the model solves the MP to obtain the optimal one.

2.6 Discretizing First-stage Decisions

The results in the previous section intuitively explain the relationship between the first and the second stage variables, and how they make optimal decisions in a collective way. By decomposing the model, obtaining the optimal solution requires evaluating possible sets of MDP parameters generated by the first stage one by one, in a discrete fashion. This provides us with incentives to consider a discretized version of the model, where all MDPs generated by the first-stage model are predefined.

As such, we consider a set of K MDP models $\mathcal{M}_k = (S, A, T_k, R_k, \gamma)$, $k = 1, \ldots, K$, representing the possible outcomes of the first-stage decision $\boldsymbol{x}$. The significance of considering multiple MDP models arises in many practical applications, where integer variables are widely utilized in modeling decision making, such as facility location, network design, vehicle routing, scheduling, etc (Conforti et al. 2014). In the literature, multiple MDP models are widely considered when MDP parameters such as the transition probability or the reward become ambiguous (Buchholz and Scheftelowitsch 2019; Steimle et al. 2021b).

Since all MDP models are predefined, the first-stage decision $\boldsymbol{x}$ no longer leads to different $\boldsymbol{\tau}$ and $\boldsymbol{r}$, but different MDP models. We further introduce a binary variable $z_k \in \{0,1\}$, $k = 1, \dots, K$ denoting the choice of the strategic decision and the following MDP models. Similar to previous models, we use a generic function $F : \mathbb{R}^n \times \{0,1\}^K \to \mathbb{R}$ to model the relationship between $\boldsymbol{x}$ and $\boldsymbol{z}$. In addition, we use the variable $V \in \mathbb{R}$ to denote the objective of the MDP selected by the variable $\boldsymbol{z}$. In the second stage, we expand the decision-making variable $y_{k,s,a}$ with an extra dimension that accounts for all K models. The K-MDP formulation with a discretized first stage, denoted by INT, is shown as follows

$$
\begin{aligned}
\text{INT} := \max \quad & \sum_{i=1}^{n} c_i x_i + V && (2.6.1)\\
\text{s.t.} \quad & \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\, j = 1, \dots, m; && (2.6.2)\\
& F(\boldsymbol{x}, \boldsymbol{z}) = 0; && (2.6.3)\\
& \sum_{k=1}^{K} z_k = 1; && (2.6.4)\\
& V \leq \sum_{s \in S} \sum_{a \in A} R_k(s,a) y_{k,s,a} + M \cdot (1 - z_k) \quad \forall\, k = 1, \dots, K; && (2.6.5)\\
& \sum_{a \in A} y_{k,s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} T_k(s'|s,a) y_{k,s',a} = \alpha_s \cdot z_k \quad \forall\, s \in S, k = 1, \dots, K; && (2.6.6)\\
& \boldsymbol{x}, \boldsymbol{y} \succeq \boldsymbol{0}, \boldsymbol{z} \in \{0,1\}^K, V \text{ unrestricted.} && (2.6.7)
\end{aligned}
$$

In the model, M denotes a very large number. It guarantees that V will take the objective value of the MDP selected by the variable $\boldsymbol{z}$. The variable z_k is also present at the Constraint (2.6.6), to ensure the feasibility of all MDP models that are not selected by $\boldsymbol{z}$.

The discretized model is formulated with mixed integer programming (MIP). Although in the essence, integer variables still bring nonlinearity to the model, there are established algorithms such as branch-and-bound to solve MIP models relatively efficiently (Conforti et al. 2014). The discretization reduces the nonlinearity of variable multiplication, since the transition probability $T_k(s'|s,a)$ and reward $R_k(s,a)$ are now parameters to a specific MDP model.

Note that by discretizing the first stage, formulation (2.6.1) – (2.6.7) is not necessarily equivalent to the original formulation (2.2.7) – (2.2.13). The two formulations are equivalent when the first-stage decisions can be naturally discretized, e.g., choosing facility sites among selected candidate locations, deciding on the number of integer resources such as personnel, or designing a network with a finite number of connectivities between nodes. Under situations where the first-stage decisions do not lead to discretized decisions, methods such as sample average approximation can still be applied to transform the model into the integer formulation (Birge and Louveaux 2011), in which case the integer formulation provides a near-optimal estimate towards the true optimal objective.

Adopting the same idea, we are also able to discretize the first stages of the two extensions to CMDP and CMDP with variable budgets. The extension to CMDP is straightforward. The following MIP model (INT-C) shows the discretized formulation of K CMDP models.

$$\text{INT-C} := \max \quad \sum_{i=1}^{n} c_i x_i + V \tag{2.6.8}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\ j = 1, \ldots, m; \tag{2.6.9}$$

$$F(\boldsymbol{x}, \boldsymbol{z}) = 0; \tag{2.6.10}$$

$$\sum_{k=1}^{K} z_k = 1; \tag{2.6.11}$$

$$V \leq \sum_{s\in S}\sum_{a\in A} R_k(s,a) y_{k,s,a} + M \cdot (1 - z_k) \quad \forall k = 1, \ldots, K; \tag{2.6.12}$$

$$\sum_{a\in A} y_{k,s,a} - \gamma \sum_{a\in A}\sum_{s'\in S} T_k(s'|s,a) y_{k,s',a} = \alpha_s \cdot z_k \quad \forall\ s \in S, k = 1, \ldots, K; \tag{2.6.13}$$

$$\sum_{s\in S}\sum_{a\in A} d_i(s,a) y_{k,s,a} \leq \bar{D}_i \quad \forall\ i \in \mathcal{D}, k = 1, \ldots, K; \tag{2.6.14}$$

$$\boldsymbol{x}, \boldsymbol{y} \succeq \boldsymbol{0}, \boldsymbol{z} \in \{0,1\}^K, V \text{ unrestricted.} \tag{2.6.15}$$

To model different budgets for different MDP models, similar to $y_{k,s,a}$, the budget variable $D_{k,i}$ is also expanded with an extra dimension that accounts for all K MDP models. The following MIP model shows the discretized formulation of K CMDP models with variable budgets (INT-VB).

$$\begin{aligned}
\text{INT-VB} := \max \quad & \sum_{i=1}^{n} c_i x_i + V && (2.6.16)\\
\text{s.t.} \quad & \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\ j = 1, \ldots, m; && (2.6.17)\\
& F(\boldsymbol{x}, \boldsymbol{z}) = 0; && (2.6.18)\\
& \sum_{k=1}^{K} z_k = 1; && (2.6.19)\\
& V \leq \sum_{s \in S} \sum_{a \in A} R_k(s,a) y_{k,s,a} - \eta \sum_{i \in \mathcal{D}} D_{k,i} + M \cdot (1 - z_k) \quad \forall k = 1, \ldots, K; && (2.6.20)\\
& \sum_{a \in A} y_{k,s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} T_k(s'|s,a) y_{k,s',a} = \alpha_s \cdot z_k \quad \forall\ s \in S, k = 1, \ldots, K; && (2.6.21)\\
& \sum_{s \in S} \sum_{a \in A} d_i(s,a) y_{k,s,a} \leq D_{k,i} \quad \forall\ i \in \mathcal{D}, k = 1, \ldots, K; && (2.6.22)\\
& D_{k,i} \leq \bar{D}_i \quad \forall\ k = 1, \ldots, K, i \in \mathcal{D}; && (2.6.23)\\
& \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{D} \succeq \boldsymbol{0}, \boldsymbol{z} \in \{0,1\}^K, V \text{ unrestricted.} && (2.6.24)
\end{aligned}$$

Although with reduced nonlinearity, the above integer formulations are still too complex to solve efficiently. The complexity mainly comes from two fronts. First, integer variables in the models require branching algorithms, whose computational costs increase exponentially with the size of the problem. Second, the curse of dimensionality in MDP models suggests large numbers of variables and constraints, rendering the models even more difficult to solve. Therefore, algorithm development is urgently required in order to solve the proposed models efficiently.

Chapter 3

Solution Algorithm

In this chapter, we develop novel algorithms with the aim to solve the models proposed in Chapter 2 more efficiently. As discussed, the main computational burden of solving the nonlinear models comes from two fronts: the nonlinearity of integer variables, and the curse of dimensionality of MDP. Since branching strategies of integer variables often depend on specific applications and problem instances, in this study, we focus on exact algorithms that solve MDP in its LP formulation. First, we propose a novel, exact solution algorithm for MDP using the Benders decomposition method. Specifically, we develop a multi-cut L-shaped (MCLD) algorithm that solves MDP iteratively. Then, we construct a two-step backward decomposition (TSBD) method that utilizes the MDP solution algorithm as a foundation to optimize the LSSD framework and its extensions.

In the literature, many studies have considered decomposition techniques for MDP to alleviate the curse of dimensionality (Daoui et al. 2010). Special structures in the Markovian transition diagram play important roles, such as the strongly communicating classes (SCC) (Ross and Varadarajan 1991; Abbad and Boustique 2003; Larach et al. 2017). Other types of decomposition involves the hierarchical structures (Bai et al. 2015), distributed optimization (Fu et al. 2015), parallel computing (Chen and Lu 2013; Chafik and Daoui 2015), or fluid optimization (Bertsimas and Mišić 2016). As such, most of the approaches decompose an MDP into smaller MDPs and solve them with dynamic programming algorithms (Abbad and Boustique 2003; Chen and Lu 2013; Larach et al. 2017), while only a few consider the decomposition of the LP formulation with exact solution methods (Kushner and Chen 1974; Dean and Lin 1995; Fu et al. 2015). Especially, the Dantzig-Wolfe decomposition (Dantzig and Wolfe 1960) was applied to the LP formulation of MDP (Kushner and Chen 1974; Dean and Lin 1995). However, since the Dantzig-Wolfe decomposition requires a block angular shape in the linear program, it only applies to MDP with special structures in the transition diagrams (Kushner and Chen 1974).

As a decomposition method closely related to Dantzig-Wolfe, the Benders decomposition (Benders 1962) has been utilized by many studies in the current literature to solve MDP-related problems (Rebennack 2016). Most of the studies feature a combination of MDP with additional constraints that model system dynamics, where the Benders decomposition is only applied to the newly added constraints, rather than the MDP itself (Dimitrov and Morton 2009; Regan and Boutilier 2012; Vickson et al. 2020; Rokhforoz and Fink 2021). The Benders decomposition has also been used to solve the recently proposed multi-model MDP (MMDP) (Steimle et al. 2021b; Steimle et al. 2021a). Specifically, the Benders method decomposes MMDP into smaller problems, each characterizing an MDP with a particular set of parameters (Steimle et al. 2021a), but each MDP is still solved as a whole. Interestingly, the generalized Benders method (Geoffrion 1972) has been adopted to derive approximate dynamic programming algorithms to solve the MDP from a reinforcement learning perspective (Warrington et al. 2019; Warrington 2019). The resulting algorithm consecutively generates lower bounds to yield approximated solutions (Warrington 2019).

To the best of our knowledge, the current literature has not seen algorithms that apply the Benders decomposition directly to solve MDP exactly. In this study, we propose a novel decomposition approach for MDP based on the Benders decomposition. The resulting Benders decomposition of MDP breaks down the LP formulation into smaller, easier-to-solve linear programs, leading to an algorithm that solves optimal policies for MDP problems much more efficiently. Different from the Dantzig-Wolfe decomposition (Kushner and Chen 1974; Dean and Lin 1995), our approach does not require a special structure in the transition diagram, making it a generalized method for all types of MDP problems. Furthermore, the

Benders decomposition of MDP provides intuitive interpretations of model variables, from which the optimal policy can be easily derived.

Figure 3.1 demonstrates the utilization of the Benders decomposition for solving MDP. The method also exploits the primal-dual relationships of the decomposed model and computes both the optimal state values as well as the optimal policy. The proposed method can be easily applied to the LSSD framework since the LP model in the extensive form remains unchanged in the decomposed model, making it advantageous compared with state-of-the-art MDP solution methods in the literature, such as modified policy iteration (MPI) (Puterman and Shin 1978) or reinforcement learning (Sutton and Barto 2018), whose dynamic programming structures prove difficult to incorporate additional linear systems. As such, the Benders decomposition of MDP not only serves as a reliable algorithm for solving MDP problems, but also as an efficient method of the LSSD framework, expanding the computational capability of the current literature for long-term strategic reasoning in complex stochastic systems.

3.1 The Decomposition of MDP

The motivation for decomposing MDP comes from the need to solve MDP in its LP formulation, where the curse of dimensionality indicates an LP model with large numbers of variables and constraints. In the literature, methods for solving large-scale linear programs have been thoroughly studied over the years. In the following, we use the well-established Benders decomposition to decompose the infinite horizon MDP and propose an L-shaped algorithm to solve the decomposed MDP efficiently. The Benders decomposition adopts a "divide-and-conquer" strategy by decomposing a large-scale linear program into an MP and multiple SPs (Benders 1962). The resulting algorithm, the L-shaped algorithm, consecutively adds constraints (cuts) to the feasible region until optimality is obtained (Birge and Louveaux 2011).

The Benders decomposition is widely used in stochastic programming problems, where the extensive form of a large-scale linear program is decomposed into smaller ones that are

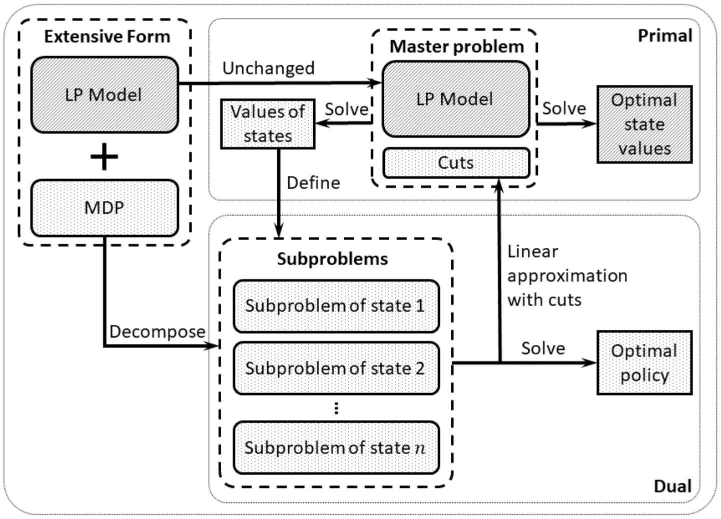


Figure 3.1: A demonstration of the proposed decomposition approach.

much easier to solve. In stochastic programming, the decomposition particularly focuses on separating possible scenarios to be realized in the future, which corresponds to the possible future states in the MDP. Since $\boldsymbol{\alpha}$ represents a probability distribution over the states, we write the objective of the LP formulation of MDP as

$$V^* = \min \sum_{s \in S} \alpha_s v_s = \min \ \mathbb{E}_s[v_s], \tag{3.1.1}$$

where $\boldsymbol{v} \in \{\boldsymbol{v} : v_s - \gamma \sum_{s' \in S} T(s'|s,a) v_{s'} \geq R(s,a), \ \forall \ s \in S, a \in A\}$. Next, we define a lower bound $\tilde{V}$ of V^*, such that

$$\tilde{V} := \min \ \mathbb{E}_s[\min \ v_s] \leq \min \ \mathbb{E}_s[v_s] = V^*, \tag{3.1.2}$$

with $\boldsymbol{v}$ defined in the same domain as in V^*. Then, $\tilde{V}$ formulates the extensive form of a stochastic programming representation of MDP:

$$\begin{aligned} \tilde{V} = \min \quad & \mathbb{E}_s[\min \ v_s] && (3.1.3)\\ \text{s.t.} \quad & v_s - \gamma \sum_{s' \in S} T(s'|s,a) v_{s'} \geq R(s,a), \ \forall \ s \in S, a \in A; && (3.1.4)\\ & v_s \text{ unrestricted}, \quad \forall \ s \in S. && (3.1.5) \end{aligned}$$

Although $\tilde{V}$ is a lower bound of V^*, we later show that the equality holds, i.e., $\tilde{V} = V^*$, so that the two formulations for MDP are equivalent.

Now we decompose the extensive form into two stages. Let $\mathcal{Q} := \mathbb{E}_s[\min \ v_s]$, the MP in the first stage becomes an unconstrained optimization problem:

$$\tilde{V} = \min \quad \mathcal{Q}, \tag{3.1.6}$$

where $\mathcal{Q} := \mathbb{E}_s\big[Q(s)\big] = \sum_s \alpha_s Q(s)$ is the expected value over all SPs in the second stage, with

$$Q(s) := \min \quad \nu_s \tag{3.1.7}$$

$$\text{s.t.} \quad \nu_s \geq R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)Q(s'), \quad \forall\ a \in A; \tag{3.1.8}$$

$$\nu_s \text{ unrestricted.} \tag{3.1.9}$$

In an SP, the variable to be optimized, $\nu_s \in \mathbb{R}$, is alone in the objective function. With constraints added for every action, each SP is equivalent to finding the tightest lower bound of ν_s using $\boldsymbol{Q}$ from the master problem. On the other hand, $Q(s)$ is also the objective value of the SP for state s, representing the value function $V(s)$. We define the variable $\boldsymbol{\theta} \in \mathbb{R}^{|S|}$ to denote the values in $\boldsymbol{Q}$. Recall that $\mathbb{E}_s\big[Q(s)\big] = \sum_{s \in S} \alpha_s Q(s)$. The MP of the decomposed MDP is thus written as follows,

$$\text{MP} := \min \quad \sum_{s \in S} \alpha_s \theta_s \tag{3.1.10}$$

$$\text{s.t.} \quad \theta_s \text{ unrestricted,} \quad \forall\ s \in S, \tag{3.1.11}$$

with each SP defined for a state $s \in S$,

$$\text{SP}(s) = \theta_s := \min \quad \nu_s \tag{3.1.12}$$

$$\text{s.t.} \quad \nu_s \geq R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'}, \quad \forall\ a \in A; \tag{3.1.13}$$

$$\nu_s \text{ unrestricted.} \tag{3.1.14}$$

Typically, in stochastic programming, two types of cuts are derived from SP and added to the MP, namely the feasibility and optimality cuts. The feasibility cuts ensure that the SP constructed with the MP solution is always feasible and the optimality cuts guarantee that the next solution is not worse than the previous one. By iteratively adding feasibility and optimality cuts, the L-shaped algorithm eventually leads to the optimal solution (Birge and Louveaux 2011). Note that for the decomposed MDP, since each SP(s) finds the tightest lower bound of v_s and v_s is unrestricted, SP(s) has complete recourse, i.e., SP is always feasible and an optimal solution to SP always exists. Thus, the following proposition holds.

Proposition 3.1. *The SP (3.1.12) – (3.1.14) has complete recourse, i.e., for all solution $\bar{\boldsymbol{\theta}}$ to the MP, there always exists a feasible solution ν_s.*

Proof. The proposition holds since the optimal solution to the SP is $\nu_s = \max\limits_{a \in A}\{R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\bar{\theta}_{s'}\}$. □

Proposition 3.1 suggests that to solve the decomposed model of MDP, feasibility cuts are unnecessary and only optimality cuts are required. To derive the optimality cuts, we first consider the dual problem (DP) of the SP. Similar to SP, DP is constructed for every $s \in S$, where each constraint in SP is associated with a dual variable. Let $\bar{\boldsymbol{\theta}}$ be the incumbent optimal solution of the MP. We define the dual variable $\boldsymbol{\mu}_s \in \mathbb{R}_+^{|A|}$ for each SP(s), $\forall\ s \in S$. Then, DP can be written as

$$\begin{aligned} \text{DP}(s) := \max \quad & \sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\bar{\theta}_{s'} \Big] \mu_{s,a} && (3.1.15) \\ \text{s.t.} \quad & \sum_{a \in A} \mu_{s,a} = 1; && (3.1.16) \\ & \mu_{s,a} \geq 0, \quad \forall\ a \in A. && (3.1.17) \end{aligned}$$

In the dual problem, with constraints (3.1.16) and (3.1.17), the objective becomes a convex combination of $R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'}$ with respect to $\boldsymbol{\mu}_s$, over the action set A. Thus, for every DP(s), we can interpret the dual variable $\boldsymbol{\mu}_s$ as a randomized policy, i.e., a probability distribution over A. Let $\bar{\boldsymbol{\mu}}_s$ be the incumbent optimal dual solution, using MP variables $\boldsymbol{\theta}$, the optimality cuts can be formulated as follows,

$$\theta_s \geq \sum_{a \in A} \bar{\mu}_{s,a} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'} \Big], \quad \forall\ s \in S. \tag{3.1.18}$$

Note that the Benders decomposition of MDP adds multiple optimality cuts at the same time, while many other stochastic programming problems add a single optimality cut at a time. The "single-cut" optimality cut summarizes the information of all future scenarios into one "θ" variable in the MP. However, for MDP, since information about the value of each state is required in every SP, the MP must maintain a vector of $\boldsymbol{\theta}$ to keep track of the state values separately, making it difficult to formulate a single-cut decomposition model.

3.2 The MCLD Algorithm

Using the optimality cuts (3.1.18), we propose the MCLD algorithm to solve the Benders decomposition of MDP. Initially, since the MP is an unconstrained optimization problem with the variable $\boldsymbol{\theta}$ unbounded, the objective value would be unbounded as well. Thus, we impose a constraint

$$\theta_s \geq -M, \quad \forall\ s \in S, \tag{3.2.1}$$

where $M \in \mathbb{R}_+$ is a sufficiently large number, such that $\boldsymbol{\theta}$ can be considered unbounded but their values are meaningful enough to avoid numerical issues in practical implementations.

Algorithm 1: The MCLD Algorithm

```
Initialize the MP variable $\theta_s$ with lower bounds $\theta_s \geq -M$, $\forall\ s \in S$, where $M$ is a very sufficiently number;
repeat
    Solve the MP and obtain solution $\bar{\theta}_s$, $\forall\ s \in S$;
    $Optimal \leftarrow$ True;
    for $s \in S$ do
        Construct DP($s$) using $\bar{\theta}_{s'}$, $\forall\ s' \in S$;
        Solve DP($s$) and obtain the solution $\bar{\mu}_{s,a}$, $\forall\ a \in A$;
        $\nu_s \leftarrow \sum_{a \in A} \left[ R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\bar{\theta}_{s'} \right] \bar{\mu}_{s,a}$;
        if $\nu_s > \bar{\theta}_s$ then
            $Optimal \leftarrow$ False;
            Add a cut (3.1.18) to MP with respect to $s$;
        else
            continue;
        end
    end
until $Optimal$;
```

After solving MP, the algorithm goes through every state $s \in S$ one by one. For a state s, first, the MP solution $\bar{\boldsymbol{\theta}}$ is used to construct DP(s). The optimality check is conducted after DP(s) is optimized, using DP(s) solutions $\bar{\boldsymbol{\mu}}_s$. The current value of a state is calculated by

$$\nu_s = \sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\bar{\theta}_{s'} \Big] \bar{\mu}_{s,a}, \tag{3.2.2}$$

which is also the optimal dual objective value. If $\theta_s < \nu_s$, an optimality cut (3.1.18) is added to MP for the state s. After going through all states, if no optimality cut is added, the current solution $\bar{\boldsymbol{\theta}}$ is optimal and the algorithm terminates. Otherwise, the MP is re-solved and the process repeats. The convergence of the L-shaped algorithm has been proved (Birge and Louveaux 2011), so the termination of the MCLD algorithm is guaranteed. The MCLD algorithm is summarized in Algorithm 1.

Suppose that Algorithm 1 terminates after L iterations, $L \in \mathbb{N}^+$, the final MP would have L sets of added optimality cuts, each consisting of at most $|S|$ cuts. Thus, the number of constraints in the MP is at most $L|S|$. Since multi-cut L-shaped algorithms typically converge within a finite number of iterations (Birge and Louveaux 2011), we conjecture $L \ll |A|$ for problems with large action spaces. Thus, the decomposed MP can be a much more compact model than the LP formulation.

The most computationally costly section in Algorithm 1 is to calculate the current value ν_s and to construct the optimality cut, which requires $|S| \cdot |A|$ operations. Since the operations are conducted for each state, the complexity for Algorithm 1 is at least $|S|^2 \cdot |A|$, without accounting for solving MP and DP. Here we add a special note for practical implementation of Algorithm 1. Observe that when calculating the current value v and constructing cuts, a part of the calculation is identical, i.e., the constant $\sum_{a \in A} R(s,a)\bar{\mu}_{s,a}$ and the coefficient of $\bar{\theta}_{s'}$ and $\theta_{s'}$. Thus, such values only need to be calculated once. Let

$$e_s = \sum_{a \in A} R(s,a)\bar{\mu}_{s,a}, \quad \forall\ s \in S \tag{3.2.3}$$

and

$$E_{s,s'} = \gamma \sum_{a \in A} T(s'|s,a)\bar{\mu}_{s,a}, \quad \forall\, s \in S, s' \in S. \tag{3.2.4}$$

The current value for state s becomes

$$\nu_s = e_s + \sum_{s' \in S} E_{s,s'}\bar{\theta}_{s'} \tag{3.2.5}$$

and the optimality cut becomes

$$\theta_s \geq e_s + \sum_{s' \in S} E_{s,s'}\theta_{s'}, \quad \forall\, s \in S. \tag{3.2.6}$$

This formulation is consistent with the multi-cut L-shaped algorithm proposed in the literature (Birge and Louveaux 2011). It avoids repeated calculations and squeezes the most computationally expensive operations into Equation (3.2.4), which we find especially effective in enhancing the performance of Algorithm 1.

Although the convenient formulation cannot exempt Algorithm 1 from the curse of dimensionality, the advantages of solving a series of smaller subproblems still strongly outweigh the benefit of solving a large linear program. Thus, the MCLD algorithm is very efficient in solving large instances of MDP. In addition, Algorithm 1 is highly modular, i.e., after solving the MP, the operations of solving DP and adding cuts can be conducted in parallel. This opens the door for parallel computing (Almasi and Gottlieb 1994), which often offers much higher computational performances than regular "serial" computing.

3.3 Mathematical Property

Now, we show mathematical properties of the decomposition method, especially its relationship with the LP formulation. First, we introduce a few new notations. Let Π_1

denote the feasible region of the LP formulation, i.e.,

$$\Pi_1 := \Big\{\boldsymbol{v} \in \mathbb{R}^{|S|} : v_s - \gamma \sum_{s' \in S} T(s'|s,a) v_{s'} \geq R(s,a), \forall\ s \in S, a \in A\Big\}, \tag{3.3.1}$$

and Π_2^ℓ the polyhedron defined by the optimality cuts added in the ℓth iteration of Algorithm 1, i.e.,

$$\Pi_2^\ell := \Big\{\boldsymbol{\theta} \in \mathbb{R}^{|S|} : \theta_s \geq \sum_{a \in A} \mu_{s,a}^\ell \big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) \theta_{s'}\big], \forall\ s \in S\Big\}, \tag{3.3.2}$$

where the MCLD algorithm runs for $\ell = 0, 1, ..., L$ iterations and $\boldsymbol{\mu}_s^\ell$, $\forall\ s \in S$, are the optimal dual variables of the ℓth iteration.

Theorem 3.1 provides a simple yet important result, which states that Π_1 is always a subset of Π_2^ℓ.

Theorem 3.1. *$\Pi_1 \subseteq \Pi_2^\ell$, $\forall\ \ell = 0, 1, \ldots, L$.*

Proof. For any $\boldsymbol{v} \in \Pi_1$, according to the definition, we have

$$v_s - \gamma \sum_{s' \in S} T(s'|s,a) v_{s'} \geq R(s,a), \quad \forall\ s \in S, a \in A. \tag{3.3.3}$$

Now, since $\mu_{s,a}^\ell \geq 0$, we take the product of $\mu_{s,a}^\ell$ with both sides,

$$v_s \mu_{s,a}^\ell - \gamma \sum_{s' \in S} T(s'|s,a) v_{s'} \mu_{s,a}^\ell \geq R(s,a) \mu_{s,a}^\ell, \quad \forall\ s \in S, a \in A. \tag{3.3.4}$$

By summing up the inequalities with respect to a, we have

$$\sum_{a \in A} v_s \mu_{s,a}^\ell - \sum_{a \in A} \gamma \sum_{s' \in S} T(s'|s,a) v_{s'} \mu_{s,a}^\ell \geq \sum_{a \in A} R(s,a) \mu_{s,a}^\ell, \quad \forall\ s \in S. \tag{3.3.5}$$

Since $\sum_{a \in A} \mu_{s,a}^\ell = 1$, $\forall s \in S$, reorganizing the inequalities,

$$v_s \geq \sum_{a \in A} \mu_{s,a}^\ell \big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) v_{s'}\big], \quad \forall\ s \in S, \tag{3.3.6}$$

which states that $\boldsymbol{v} \in \Pi_2^\ell$. Because $\mu_{s,a}^\ell \geq 0$ and $\sum_{a\in A} \mu_{s,a}^\ell = 1$ is true for all $\ell = 0, 1, \ldots, L$, the result holds. □

Corollary 3.1.1, 3.1.2 and 3.1.3 are extensions to Theorem 3.1. Corollary 3.1.1 states that Π_1 is a subset of the feasible region of the MP at the ℓth iteration, for every $\ell = 0, 1, \ldots, L$.

Corollary 3.1.1. $\Pi_1 \subseteq \bigcap_{j=0}^{\ell} \Pi_2^j$, $\forall\ \ell = 0, 1, \ldots, L$.

Proof. Since $\Pi_1 \subseteq \Pi_2^\ell$, $\forall\ \ell = 0, 1, \ldots, L$, it is natural that $\Pi_1 \subseteq \bigcap_{j=0}^{\ell} \Pi_2^j$. □

Corollary 3.1.2 states the existence of the optimal solution to the decompose MDP. Since it has been proven that the optimal solution to the infinite horizon MDP always exits (Howard 1960), Corollary 3.1.2 shows that the optimal solution to the decomposed MDP exists as well.

Corollary 3.1.2. *If* Π_1 *is non-empty,* $\bigcap_{j=0}^{\ell} \Pi_2^j$ *is non-empty,* $\forall\ \ell = 0, 1, \ldots, L$.

Proof. This is a direct result of Corollary 3.1.1, that $\Pi_1 \subseteq \bigcap_{j=0}^{\ell} \Pi_2^j$, $\forall\ \ell = 0, 1, \ldots, L$. □

Corollary 3.1.3 shows that at each iteration, after optimality cuts are added, the MP computes a lower bound to the optimal value V^*. Moreover, the lower bound improves as more cuts are added.

Corollary 3.1.3. *Let*

$$V^* = \min\ \boldsymbol{\alpha} \cdot \boldsymbol{v},\ \ \boldsymbol{v} \in \Pi_1, \tag{3.3.7}$$

and

$$\tilde{V}^\ell = \min\ \boldsymbol{\alpha} \cdot \boldsymbol{\theta},\ \ \boldsymbol{\theta} \in \bigcap_{j=0}^{\ell} \Pi_2^j,\ \ \forall\ \ell = 0, 1, \ldots, L. \tag{3.3.8}$$

Then, $V^* \geq \tilde{V}^L \geq \tilde{V}^{L-1} \geq \cdots \geq \tilde{V}^0$.

Proof. The relationship $\tilde{V}^L \geq \tilde{V}^{L-1} \geq \cdots \geq \tilde{V}^0$ comes from the fact that $\bigcap_{j=0}^{\ell+1} \Pi_2^j \subseteq \bigcap_{j=0}^{\ell} \Pi_2^j$, $\forall\ \ell = 0, 1, \ldots, L$, since each feasible region of MP contains all previous optimality cuts and new cuts are added on top of those. Moreover, $V^* \geq \tilde{V}^\ell$, $\forall\ \ell = 0, 1, ..., L$ because of Corollary 3.1.1, that $\Pi_1 \subseteq \bigcap_{j=0}^{\ell} \Pi_2^j$, $\forall\ \ell = 0, 1, \ldots, L$. □

Corollary 3.1.3 provides more insights regarding how Algorithm 1 operates. Through continuously adding optimality cuts to the MP, Algorithm 1 finds better and better lower bounds to the true optimal value V^*, until algorithm termination. However, Corollary 3.1.3 does not necessarily indicate that after termination, the value of $\tilde{V}^L$ converges to V^*. Note that at the Lth (final) iteration, $\tilde{V}^L = \tilde{V}$ defined in Equation (3.1.2).

With the proof of convergence, we show that the equality is achieved in Equation (3.1.2), and the decomposed MDP model is equivalent to the LP formulation. In Theorem 3.2, we show that the termination condition of Algorithm 1 leads to the convergence of $\tilde{V}^\ell$ towards V^*, proving that the MCLD algorithm solves MDP exactly.

Theorem 3.2. *$V^* = \tilde{V}^L$.*

Proof. Let $\boldsymbol{v}^*$ corresponds to the optimal solution to V^* and $\boldsymbol{\theta}^*$ the solution to $\tilde{V}^L$. The termination condition of Algorithm 1 states that

$$\theta_s^L \geq \sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'}^L \Big] \mu_{s,a}^L, \quad \forall\ s \in S. \tag{3.3.9}$$

Since $\mu_{s,a}^L$ are the dual variables representing a convex combination that maximizes the dual objective $\sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'}^L \Big] \mu_{s,a}^L$, we have

$$\begin{aligned} \theta_s^L \geq & \sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'}^L \Big] \mu_{s,a}^L \\ = & \max_{a \in A} \Big\{ R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'}^L \Big\}, \quad \forall\ s \in S, \end{aligned} \tag{3.3.10}$$

which suggests

$$\theta_s^L \geq R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'}^L, \quad \forall\ s \in S, a \in A. \tag{3.3.11}$$

Thus, $\boldsymbol{\theta}^L$ is a feasible solution to the LP formulation, i.e., $\boldsymbol{\theta}^L \in \Pi_1$. This means that $\boldsymbol{\alpha}^T \cdot \boldsymbol{v}^* \leq \boldsymbol{\alpha}^T \cdot \boldsymbol{\theta}^L$. However, from Corollary 3.1.3, we know $\boldsymbol{\alpha}^T \cdot \boldsymbol{v}^* \geq \boldsymbol{\alpha}^T \cdot \boldsymbol{\theta}^L$. Therefore, we conclude that $V^* = \boldsymbol{\alpha}^T \cdot \boldsymbol{v}^* = \boldsymbol{\alpha}^T \cdot \boldsymbol{\theta}^L = \tilde{V}^L$. □

Let $\boldsymbol{v}^*$ be the optimal solution to the LP formulation. Corollary 3.2.1 shows a direct results of Theorem 3.2, that $\boldsymbol{v}^*$ and $\boldsymbol{\theta}^L$ are equal.

Corollary 3.2.1. *$\boldsymbol{v}^* = \boldsymbol{\theta}^L$.*

Next, Theorem 3.3 shows the derivation of deterministic and randomized optimal policies using the dual variable $\boldsymbol{\mu}_s^L$, for each state $s \in S$.

Theorem 3.3. *Let $\pi^* : S \to A$ be an optimal policy. Then,*

(a) $\pi^*(s) = \arg\max_{a \in A} \mu_{s,a}^L$ characterizes a deterministic optimal policy;

(b) $Pr\Big\{\pi^*(s) = a\Big\} = \mu_{s,a}^L$ characterizes a randomized optimal policy, where $Pr\{\cdot\}$ is the probability of choosing an action a under state s.

Proof. From Corollary 3.2.1, we have $\boldsymbol{v}^* = \boldsymbol{\theta}^L$. Thus, $\theta_s^L = V^*(s)$, $\forall\ s \in S$. Since $\boldsymbol{\mu}_s^L \succeq \mathbf{0}$ and $\boldsymbol{\mu}_s^L \cdot \mathbf{1} = 1$, at optimum, the objective of DP(s) becomes

$$\begin{aligned}\sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)V^*(s')\Big]\mu_{s,a}^L \\ =\max_{a \in A}\Big\{R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)V^*(s')\Big\} = V^*(s),\end{aligned} \tag{3.3.12}$$

suggesting that $\boldsymbol{\mu}_s^L$ is a probability distribution over A that maximizes the optimality equation. Part (b) follows naturally that $\boldsymbol{\mu}_s^L$ is the randomized optimal policy.

To prove part (a), suppose there exists $a_1, a_2 \in A$ such that $\mu^L_{s,a_1} \leq \mu^L_{s,a_2}$, but $R(s,a_1) + \gamma \sum_{s' \in S} T(s'|s,a_1)V^*(s') > R(s,a_2) + \gamma \sum_{s' \in S} T(s'|s,a_2)V^*(s')$. Then,

$$\begin{aligned}
&\big[R(s,a_1) + \gamma \sum_{s' \in S} T(s'|s,a_1)V^*(s')\big] \cdot \mu^L_{s,a_2} \\
&\qquad > \big[R(s,a_2) + \gamma \sum_{s' \in S} T(s'|s,a_2)V^*(s')\big] \cdot \mu^L_{s,a_1},
\end{aligned} \tag{3.3.13}$$

such that the original objective $\sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)V^*(s')\Big] \mu^L_{s,a}$ is no longer optimal, leading to a contradiction. Thus, $\forall\ a \in A$, $\mu^L_{s,a_1} \leq \mu^L_{s,a_2}$ suggests $R(s,a_1) + \gamma \sum_{s' \in S} T(s'|s,a_1)V^*(s') \leq R(s,a_2) + \gamma \sum_{s' \in S} T(s'|s,a_2)V^*(s')$ and

$$\pi^*(s) = \arg\max_{a \in A} \mu^L_{s,a} = \arg\max_{a \in A} \Big\{R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)V^*(s')\Big\}. \tag{3.3.14}$$

This completes the proof. □

In Theorem 3.3, the dual variables $\boldsymbol{\mu}^L_s$ characterizes both the deterministic and randomized optimal policy of the MDP. By defining $\boldsymbol{\mu}^L_s$ in DP(s) for each state s as binary variables, we can compute the deterministic optimal policy. Similarly, by defining $\boldsymbol{\mu}^L_s$ as continuous variables, we can compute the randomized optimal policy. The equivalence of deterministic and randomized policy can be shown through a special property of DP(s). Note that the left-hand-side parameter matrix of DP(s) is totally unimodular, i.e., the square submatrices all have determinants 0, 1 or -1 (Conforti et al. 2014). The totally unimodular property makes DP(s) a perfect formulation, where the integer program has the same optimal solution as the relaxed linear program (Conforti et al. 2014). Thus, the deterministic policy, where $\boldsymbol{\mu}^L_s$ are binary variables, is equivalent to the randomized policy, where $\boldsymbol{\mu}^L_s$ are continuous variables.

3.4 Special MDP

In the following, we investigate the Benders decomposition approach for special MDPs. First, we consider MDP with monotone optimal policy. Then, we consider MDP with additional constraints. For both classes of MDP, we modify the MCLD algorithm by developing specialized optimality cuts. In addition, we include the theoretical results of another special type MDP in Appendix 3.4.1, where the transition probabilities are independent of actions.

3.4.1 Action-free Transition Probability

Here, we focus on a special MDP where the transition probability does not depend on the actions, i.e., $T(s'|s,a) = T(s'|s)$, $\forall\ a \in A$. This indicates that the stochastic environment, captured by the states, transitions independently at every decision epoch. Thus, the LP formulation, MP, SP and DP for this special case can be written by simply substituting $T(s'|s,a)$ for $T(s'|s)$. We define $\hat{V}$ as the objective value of the following program:

$$\begin{aligned}
\hat{V} := \min \quad & \sum_{s\in S} \alpha_s \theta_s && (3.4.1)\\
\text{s.t.} \quad & \theta_s \geq \sum_{a\in A} \bar{\mu}_{s,a}\Big[R(s,a) + \gamma \sum_{s'\in S} T(s'|s)\theta_{s'}\Big], \quad \forall\ s \in S, && (3.4.2)\\
& \theta_s \text{ unrestricted}, \quad \forall\ s \in S, && (3.4.3)
\end{aligned}$$

where $\bar{\boldsymbol{\mu}}_s$ are the solutions to DP(s) for all $s \in S$, using the $\bar{\boldsymbol{\theta}}$ from the first iteration in the MCLD algorithm. Next, in Theorem 3.4, we show that $\tilde{V}$ is actually the optimal value of the LP formulation and the formulation (3.4.1) – (3.4.3) is equivalent to the LP formulation.

Theorem 3.4. $\hat{V} = V^*$.

Proof. First note that the objective functions are identical for $\hat{V}$ and V^*. To show the equivalence, we only need to show that they share the same feasible region. In the LP formulation, since the transition probability is independent of the actions, we have

$$v_s - \gamma \sum_{s' \in S} T(s'|s) v_{s'} \geq R(s,a), \quad \forall\ s \in S, a \in A, \tag{3.4.4}$$

which is equivalent to

$$v_s - \gamma \sum_{s' \in S} T(s'|s) v_{s'} \geq \max_{a \in A} R(s,a), \quad \forall\ s \in S. \tag{3.4.5}$$

Now consider the formulation (3.4.1) – (3.4.3). In the MCLD algorithm, since we impose the constraints $\theta_s \geq -M$, $\forall\ s \in S$, at $k = 0$, we have $\bar{\theta}_s = -M$, $\forall\ s \in S$. Thus, the objective of DP(s) becomes

$$\begin{aligned} &\max \sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) \bar{\theta}_{s'} \Big] \mu_{s,a} \\ =&\max \sum_{a \in A} R(s,a) \mu_{s,a} + \gamma \sum_{a \in A} \sum_{s' \in S} T(s'|s)(-M) \mu_{s,a} \\ =&- M \cdot \gamma + \max \sum_{a \in A} R(s,a) \mu_{s,a}, \end{aligned} \tag{3.4.6}$$

where the last equal sign holds because $\sum_{s' \in S} T(s'|s) = 1$, $\sum_{a \in A} \mu_{s,a} = 1$ and $-M \cdot \gamma$ is independent of s and a. Since $-M \cdot \gamma$ is a constant, the optimal $\bar{\boldsymbol{\mu}}_s$ are those that maximizes $R(s,a)$, $\forall\ a \in A$. As such, constraint (3.4.2) becomes

$$\begin{aligned} \theta_s \ &\geq \sum_{a \in A} \bar{\mu}_{s,a} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s) \theta_{s'} \Big] \\ &\geq \sum_{a \in A} \bar{\mu}_{s,a} R(s,a) + \gamma \sum_{a \in A} \bar{\mu}_{s,a} \sum_{s' \in S} T(s'|s) \theta_{s'} \\ &= \max_{a \in A} R(s,a) + \gamma \sum_{s' \in S} T(s'|s) \theta_{s'}, \quad \forall\ s \in S \end{aligned} \tag{3.4.7}$$

$$\Leftrightarrow \quad \theta_s - \gamma \sum_{s' \in S} T(s'|s) \theta_{s'} \geq \max_{a \in A} R(s,a), \quad \forall\ s \in S, \tag{3.4.8}$$

which is equivalent to the constraint (3.4.5) of the LP formulation. Therefore, the program (3.4.1) – (3.4.3) is equivalent to the LP formulation, hence $\hat{V} = V^*$. □

Theorem 3.4 suggests that when the transition probability is independent of the actions, Algorithm 1 would converge at the second iteration, i.e., $L = 1$. Thus, in this special case, Algorithm 1 offers a simple yet efficient way to obtain the optimal policy through decomposing the MDP.

3.4.2 Monotone Optimal Policy

The monotone optimal policy is an important structural property existing in many MDP. We consider an MDP in which n states and m actions can be ordered in such a way that

$$s_1 \leq s_2 \leq \cdots \leq s_n \quad \text{and} \quad a_1 \leq a_2 \leq \cdots \leq a_m.$$

Then, a monotone optimal policy indicates an optimal policy π^* non-decreasing in s, i.e., for any $s_i \leq s_j$, $\pi^*(s_i) \leq \pi^*(s_j)$. In the literature, many have proved sufficient conditions to the existence of a monotone policy (Puterman 2014; Krishnamurthy 2016). Let $\tau(s'|s,a)$ be the tail-sum of the transition probability, i.e., $\tau(s'|s,a) = \sum_{i=s'}^{s_n} T(i|s,a)$. If the following holds,

1. $R(s,a)$ is non-decreasing in s, for all $a \in A$;
2. $\tau(s'|s,a)$ is non-decreasing in s, for all $s' \in S$ and $a \in A$;
3. $R(s,a)$ is a superadditive (supermodular) function on $S \times A$;
4. $\tau(s'|s,a)$ is a superadditive function on $S \times A$, for all $s' \in S$,

a monotone optimal policy is guaranteed to exist (Puterman 2014). Monotone optimal policies have been observed in many applications of MDP (Alagoz et al. 2007; Shi et al. 2019; Asadi and Pinkley 2021). More importantly, they allow decision makers to derive faster algorithms for finding the optimal policy and provide intuitive insights to interpret the optimal policy (Zhuang and Li 2012; Mattila et al. 2017).

Recall that in Theorem 3.3, the optimal dual variables $\boldsymbol{\mu}_s^L$ characterize the optimal policy, which allows the monotone optimal policy to be represented in terms of $\boldsymbol{\mu}_s^L$, i.e., for all $s_i \leq s_j$,

$$\arg\max_{a\in A}\ \mu^L_{s_i,a} = \pi^*(s_i) \leq \pi^*(s_j) = \arg\max_{a\in A}\ y^L_{s_j,a}. \tag{3.4.9}$$

Let $a_i^* = \arg\max\limits_{a\in A}\ \mu^L_{s_i,a}$. Then, the above inequality suggests that for all $s_j \geq s_i$,

$$\mu^L_{s_j,a} \leq \mu^L_{s_j,a_i^*}, \quad \forall\ a \in A, a < a_i^*. \tag{3.4.10}$$

In the following, we show that the above relationship can be exploited in the MCLD algorithms to eliminate suboptimal actions. Specifically, we impose a constraint that forces some of the dual variables to be zero. Let $\bar{a}_{i-1}$ is the best action for a state s_{i-1}, i.e.,

$$\bar{a}_{i-1} = \begin{cases} a_1 & i = 1, \\ \arg\max\limits_{a\in A}\ \mu_{s_{i-1},a}, & 2 \leq i \leq m. \end{cases} \tag{3.4.11}$$

Then, we define the dual of SP(s_i) for MDP with monotone optimal policy as follows:

$$\begin{aligned} \mathrm{DP}_{\mathrm{mono}}(s_i) := \max \quad & \sum_{a\geq \bar{a}_{i-1}} \Big[R(s_i,a) + \gamma \sum_{s'\in S} T(s'|s_i,a)\bar{\theta}_{s'}\Big]\mu_{s_i,a} && (3.4.12)\\ \text{s.t.} \quad & \sum_{a\geq \bar{a}_{i-1}} \mu_{s,a} = 1; && (3.4.13)\\ & \mu_{s,a} = 0, \quad \forall\ a \in A, a < \bar{a}_{i-1}; && (3.4.14)\\ & \mu_{s,a} \geq 0, \quad \forall\ a \in A, a \geq \bar{a}_{i-1}. && (3.4.15) \end{aligned}$$

As such, at the ℓth iteration of the MCLD algorithm, we generate cuts of the following form:

$$\theta_s \geq \sum_{a\geq \bar{a}^{\ell}_{i-1}} \bar{\mu}^{\ell}_{s_i,a}\Big[R(s_i,a) + \gamma \sum_{s'\in S} T(s'|s_i,a)\theta_{s'}\Big], \quad \forall\ s_i \in S. \tag{3.4.16}$$

Note that by adding cuts (3.4.16), the MCLD algorithm still converges to the true optimal value. Since $\boldsymbol{\mu}_s^L$ defines a convex combination over the actions, letting some of the $\mu_{s,a}^L = 0$ in $\mathrm{DP}_{\mathrm{mono}}(s)$ shall not affect the optimum, as long as the maximum among $R(s_i, a) + \gamma \sum_{s' \in S} T(s'|s_i, a)\bar{\theta}_{s'}$ corresponds to a non-zero coefficient. This can be easily illustrated by deriving a distribution $\boldsymbol{\mu'}_s^L$ for a "deterministic policy" from a randomized policy distribution $\boldsymbol{\mu}_s^L$, i.e.,

$$\mu'^L_{s,a} = \begin{cases} 1, & a = \arg\max_{a \in A} \mu_{s,a}^L, \\ 0, & \text{otherwise.} \end{cases} \tag{3.4.17}$$

Thus, the optimal policy π^* produced by adding cuts (3.4.16) can be viewed as the combination of a deterministic and a randomized policy, where $Pr\Big\{\pi^*(s) = a\Big\} > 0$ only when $a \geq \bar{a}_{i-1}^L$. Asymptotically, π^* will results in the same long-term rewards as the deterministic or randomized policy calculated using the original MCLD algorithm.

3.4.3 CMDP

Now, we consider the decomposition of CMDP through its primal formulation. Let v_s denote the primal variable corresponding to constraint (2.3.2) and ρ_i the primal variable corresponding to constraints (2.3.3). The primal form can be written as

$$\begin{aligned} \min \quad & \sum_{s \in S} \alpha_s v_s + \sum_{i \in \mathcal{D}} D_i \rho_i && (3.4.18) \\ \text{s.t.} \quad & v_s - \gamma \sum_{s' \in S} T(s'|s, a) v_{s'} + \sum_{i \in \mathcal{D}} d(s, a)\rho_i \geq R(s, a), \quad \forall\ s \in S, a \in A; && (3.4.19) \\ & v_s \text{ unrestricted}, \quad \forall\ s \in S; && (3.4.20) \\ & \rho_i \geq 0, \quad \forall\ i \in \mathcal{D}. && (3.4.21) \end{aligned}$$

Based on the above primal formulation, we present the following decomposition of CMDP, as an extension to the MP and SP in Section 3.1. Similarly, we let $\boldsymbol{\theta}$ be the approximation of state values. The master problem is

$$
\begin{aligned}
\mathrm{MP}_C := \min \quad & \sum_{s \in S} \alpha_s \theta_s + \sum_{i \in \mathcal{D}} D_i \rho_i && (3.4.22)\\
\text{s.t.} \quad & \theta_s \text{ unrestricted}, \quad \forall\, s \in S; && (3.4.23)\\
& \rho_i \geq 0, \quad \forall\, i \in \mathcal{D}, && (3.4.24)
\end{aligned}
$$

with the subproblems defined for each $s \in S$,

$$
\begin{aligned}
\mathrm{SP}_C(s) = \theta_s := \min \quad & \nu_s && (3.4.25)\\
\text{s.t.} \quad & \nu_s \geq R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\bar{\theta}_{s'} \\
& \qquad - \sum_{i \in \mathcal{D}} d_i(s,a)\bar{\rho}_i, \quad \forall\, a \in A; && (3.4.26)\\
& \nu_s \text{ unrestricted}, && (3.4.27)
\end{aligned}
$$

where ν_s is the variable for $\mathrm{SP}_C(s)$ and $\bar{\boldsymbol{\theta}}$, $\bar{\boldsymbol{\rho}}$ are the optimal values calculated by MP_C. Note that adding the variable $\boldsymbol{\rho}$ does not affect the optimality of the decomposition, since $\boldsymbol{\rho}$ is treated as a master problem variable in an ordinary two-stage stochastic program (Birge and Louveaux 2011).

Now, we define $\mu_{s,a}$ as the dual variable to $\mathrm{SP}_C(s)$. Then the dual of $\mathrm{SP}_C(s)$ is

$$
\begin{aligned}
\mathrm{DP}_C(s) := \max \quad & \sum_{a \in A} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\bar{\theta}_{s'} - \sum_{i \in \mathcal{D}} d_i(s,a)\bar{\rho}_i \Big] \mu_{s,a} && (3.4.28)\\
\text{s.t.} \quad & \sum_{a \in A} \mu_{s,a} = 1; && (3.4.29)\\
& \mu_{s,a} \geq 0, \quad \forall\, a \in A. && (3.4.30)
\end{aligned}
$$

Note that the dual problem still maintains complete recourse, since the new constraint (3.4.26) in the primal problem imposes no impact on the constraints in the dual problem.

From $\text{DP}_C(s)$, we generate the following optimality cuts

$$\theta_s \geq \sum_{a \in A} \bar{\mu}_{s,a} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a)\theta_{s'} - \sum_{i \in \mathcal{D}} d_i(s,a)\rho_i \Big], \quad \forall\, s \in S, \tag{3.4.31}$$

where $\bar{\boldsymbol{\mu}}_s$ are optimal dual variables for each SP and $\boldsymbol{\theta}$, $\boldsymbol{\rho}$ are variables of the MP. Thus, we can easily apply the MCLD algorithm to solve CMDP. In stead of adding cuts (3.1.18), we add cuts (3.4.31) iteratively until the termination condition is met.

3.5 The TSBD Method

In this section, we apply the MCLD algorithm to solve the LSSD framework introduced in Chapter 2. Specifically, we develop the TSBD method aiming to solve integer models, i.e., INT, INT-C, and INT-VB, where nonlinearity is reduced through the alternate formulations.

3.5.1 INT

In the INT model, K MDP models are solved and evaluated in order to determine the optimal strategic decision and future operations. To apply the MCLD algorithm to the INT model, we consider the method. In Step-I, we evaluate all K MDP models in the second stage of the model using the decomposition method. In Step-II, we turn backwards to the first stage and use the results obtained in Step-I to construct a mixed integer programming (MIP) model to solve the optimal strategic decision in the first stage.

In Step-I, it would be computationally inefficient if we evaluate all K MDP models in a sequential manner. Thus, we utilize the iterative behavior of Algorithm 1 and evaluates K MDP models at the same time. We name the algorithm K-MCLD algorithm. Specifically, at the beginning of the algorithm, we construct an MP consisting of the objectives of all K MDP models.

$$\min \quad \sum_{k=1}^{K} \sum_{s \in S} \alpha_s \theta_{k,s} \tag{3.5.1}$$

$$\text{s.t.} \quad \theta_{k,s} \text{ unrestricted} \quad \forall\, s \in S, k = 1, \ldots, K. \tag{3.5.2}$$

Then, at each iteration ℓ, we formulate and solve the DP for each model k, state s using $\bar{\boldsymbol{\theta}}$ from the MP.

$$\max \quad \sum_{a \in A} \Big[R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \bar{\theta}_{k,s'} \Big] \mu_{k,s,a} \tag{3.5.3}$$

$$\text{s.t.} \quad \sum_{a \in A} \mu_{k,s,a} = 1; \tag{3.5.4}$$

$$\mu_{k,s,a} \geq 0 \quad \forall\, a \in A. \tag{3.5.5}$$

After obtaining the optimal solution $\bar{\boldsymbol{\mu}}$, we check the convergence condition using the variable $\nu_{k,s}$, where

$$\nu_{k,s} = \sum_{a \in A} \Big[R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \bar{\theta}_{k,s'} \Big] \bar{\mu}_{k,s,a}. \tag{3.5.6}$$

We stop solving the SP with respect to model k, state s, if $\nu_{k,s} \leq \bar{\theta}_{k,s}$. Otherwise the following optimality cut is added to the MP for model k, state s.

$$\theta_{k,s} \geq \sum_{a \in A} \bar{\mu}_{k,s,a} \Big[R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \theta_{k,s'} \Big] \quad \forall\, s \in S, k = 1, \ldots K. \tag{3.5.7}$$

Note that in the above procedure, the variables $\theta_{k,s}$ and $\mu_{k,s,a}$ are expanded with an additional dimension for the kth MDP model. The K-MCLD algorithm is shown in Algorithm 2 in detail.

In Step-II, we use the results obtained in Step-I to solve the optimal strategic decision in the first stage of the framework. To do so, first recall Corollary 3.1.3, where we show that among all optimality cuts added to the MP throughout the iterations, only those added at the Lth (final) iteration are the most binding ones. Thus, we are able to use the optimality cuts at the Lth iteration as "linear approximators" to the state values of the MDP models

Algorithm 2: The K-MCLD algorithm in Step-I.

```
Initialize $\theta_{k,s}$ with lower bounds $\theta_{k,s} \geq -M$, $\forall\ s \in S, k = 1, \ldots, K$, where $M$ is a
  very large number;
Initialize $Converged_k \leftarrow$ False, $\forall\ k = 1, \ldots, K$;
repeat
    Solve the MP (3.5.1) – (3.5.2) and obtain solution $\bar{\theta}_{k,s}$, $\forall\ s \in S, k = 1, \ldots, K$;
    $Optimal \leftarrow$ True;
    for $k = 1, \ldots, K$ do
        if $Converged_k$ then
            continue;
        end
        $Converged_k \leftarrow$ True;
        for $s \in S$ do
            Construct DP$(k, s)$ (3.5.3) – (3.5.5) using $\bar{\theta}_{k,s'}$, $\forall\ s' \in S$;
            Solve DP$(k, s)$ and obtain the solution $\bar{\mu}_{k,s,a}$, $\forall\ a \in A$;
            $\nu_{k,s} \leftarrow \sum_{a \in A} \left[ R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \bar{\theta}_{k,s'} \right] \bar{\mu}_{k,s,a}$;
            if $\nu_{k,s} > \bar{\theta}_{k,s}$ then
                $Converged_k \leftarrow$ False, $Optimal \leftarrow$ False;
                Add a cut (3.5.7) to MP with respect to $k$ and $s$;
            else
                continue;
            end
        end
    end
    $\mu^*_{k,s,a} \leftarrow \bar{\mu}_{k,s,a}$;
until $Optimal$;
```

(Birge and Louveaux 2011). It is guaranteed that the linear approximations compute the optimal objective values to MDP models thanks to Theorem 3.2.

In order to formulate the optimality cuts, we record the optimal dual variables $\boldsymbol{\mu}^*$ from Algorithm 2. Then, the following MIP can be formulated to calculate the optimal strategic decisions in the first stage.

$$\max \quad \sum_{i=1}^{n} c_i x_i + V \tag{3.5.8}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} w_{j,i} x_i = b_j \quad \forall\ j = 1, \ldots, m; \tag{3.5.9}$$

$$F(\boldsymbol{x}, \boldsymbol{z}) = 0; \tag{3.5.10}$$

$$V \le \sum_{s\in S} \alpha_s \theta_{k,s} + M \cdot (1 - z_k) \quad \forall k = 1, \dots, K; \tag{3.5.11}$$

$$\theta_{k,s} \le \sum_{a\in A} \mu^*_{k,s,a} \Big[R_k(s,a) + \gamma \sum_{s'\in S} T_k(s'|s,a)\theta_{k,s'} \Big] \quad \forall\ s \in S, k = 1, \dots, K; \tag{3.5.12}$$

$$\boldsymbol{x} \succeq \mathbf{0}, \boldsymbol{z} \in \{0,1\}^K, \boldsymbol{\theta}, V \text{ unrestricted.} \tag{3.5.13}$$

Note that different from previous optimality cuts, Constraint (3.5.12) uses "$\le$" rather than "$\ge$", because the overall objective of the MIP is maximizing.

Then, the TSBD method can be summarized as follows

- Step-I: Obtain $\boldsymbol{\mu}^*$ using Algorithm 2;
- Step-II: Solve the MIP model (3.5.8) – (3.5.13) for the optimal solution to $\boldsymbol{x}$.

3.5.2 INT-C & INT-VB

Extending the TSBD method to solve INT-C (TSBD-C) is relatively straightforward. Since the additional linear constraints of CMDP are incorporated into the second stage, Step-II of the method mostly remains the same. Step-I of the method requires adjustment according to the results established in Chapter 3.4.3, where the decomposition method for CMDP is developed.

Specifically, we construct the following MP in Step-I.

$$\min \quad \sum_{k=1}^{K} \sum_{s\in S} \alpha_s \theta_{k,s} + \sum_{k=1}^{K} \sum_{i\in\mathcal{D}} \rho_{k,i} \bar{D}_i \tag{3.5.14}$$

$$\text{s.t.} \quad \boldsymbol{\rho} \succeq \mathbf{0}, \boldsymbol{\theta} \text{ unrestricted.} \tag{3.5.15}$$

At each iteration, we formulation the following DP for model k, state s using $\bar{\boldsymbol{\theta}}$ and $\bar{\boldsymbol{\rho}}$ obtained from MP.

$$\max \quad \sum_{a\in A} \Big[R_k(s,a) + \gamma \sum_{s'\in S} T_k(s'|s,a)\bar{\theta}_{k,s'} - \sum_{i\in\mathcal{D}} d_i(s,a)\bar{\rho}_{k,i} \Big] \mu_{k,s,a} \tag{3.5.16}$$

$$\text{s.t.} \quad \sum_{a \in A} \mu_{k,s,a} = 1; \tag{3.5.17}$$

$$\mu_{k,s,a} \geq 0 \quad \forall\ a \in A. \tag{3.5.18}$$

Then, using $\bar{\mu}$ from DP, the value of convergence is modified as

$$\nu_{k,s} = \sum_{a \in A} \bar{\mu}_{k,s,a} \Big[R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \bar{\theta}_{k,s'} - \sum_{i \in \mathcal{D}} d_i(s,a) \bar{\rho}_{k,i} \Big], \tag{3.5.19}$$

and is still compared with $\bar{\theta}_{k,s}$ to determine the optimality. If necessary, optimality cuts of the following form are added back to the MP.

$$\theta_{k,s} \geq \sum_{a \in A} \bar{\mu}_{k,s,a} \Big[R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \theta_{k,s'} - \sum_{i \in \mathcal{D}} d_i(s,a) \rho_{k,i} \Big] \quad \forall\ s \in S, k = 1, \ldots K. \tag{3.5.20}$$

Finally, constraints (3.5.12) in the integer model in Step-II is modified as

$$\begin{aligned} \theta_{k,s} \leq \sum_{a \in A} \mu^*_{k,s,a} \Big[R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \theta_{k,s'} \\ - \sum_{i \in \mathcal{D}} d_i(s,a) \rho^*_{k,i} \Big] \quad \forall\ s \in S, k = 1, \ldots, K, \end{aligned} \tag{3.5.21}$$

where $\boldsymbol{\mu}^*$ and $\boldsymbol{\rho}^*$ are optimal values of variables from Step-I.

Similarly, we can extend the TSBD method to solve INT-VB (TSBD-VB). The model with variable budgets takes a different form from CMDP. Thus, we derive its decomposition by considering the following formulation, where the variable $D_{k,i}$ and weight parameter η are introduced into CMDP as the variable budgets.

$$\max \quad \sum_{s \in S} \sum_{a \in A} R_k(s,a) y_{k,s,a} - \eta \sum_{i \in \mathcal{D}} D_{k,i} \tag{3.5.22}$$

$$\text{s.t.} \quad \sum_{a \in A} y_{k,s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} T_k(s|s',a) y_{k,s',a} = \alpha_s \quad \forall\ s \in S; \tag{3.5.23}$$

$$\sum_{s \in S} \sum_{a \in A} d_i(s,a) y_{k,s,a} \leq D_{k,i} \quad \forall\, i \in \mathcal{D}; \tag{3.5.24}$$

$$D_{k,i} \leq \bar{D}_i \quad \forall\, i \in \mathcal{D}; \tag{3.5.25}$$

$$\boldsymbol{y}, \boldsymbol{D} \succeq \boldsymbol{0}. \tag{3.5.26}$$

To decompose the model, we first take the primal form of the above formulation, where $\boldsymbol{v}$, $\boldsymbol{\rho}$, and $\boldsymbol{\lambda}$ are the corresponding dual variables.

$$\max \quad \sum_{s \in S} \alpha_s v_{k,s} + \sum_{i \in \mathcal{D}} \bar{D}_i \lambda_{k,i} \tag{3.5.27}$$

$$\text{s.t.} \quad v_{k,s} - \gamma \sum_{s' \in S} T_k(s'|s,a) v_{k,s'} + \sum_{i \in \mathcal{D}} d_i(s,a) \rho_{k,i} \geq R_k(s,a) \quad \forall\, s \in S, a \in A \tag{3.5.28}$$

$$- \rho_{k,i} + \lambda_{k,i} \geq -\eta \quad \forall\, i \in \mathcal{D}; \tag{3.5.29}$$

$$\boldsymbol{\rho}, \boldsymbol{\lambda} \succeq \boldsymbol{0}, \boldsymbol{v} \text{ unrestricted.} \tag{3.5.30}$$

Next, in Step-I, we formulate the MP used in Algorithm 2 for INT-VB

$$\min \quad \sum_{k=1}^{K} \sum_{s \in S} \alpha_s \theta_{k,s} + \sum_{k=1}^{K} \sum_{i \in \mathcal{D}} \bar{D}_i \lambda_{k,i} \tag{3.5.31}$$

$$\text{s.t.} \quad - \rho_{k,i} + \lambda_{k,i} \geq -\eta \quad \forall\, i \in \mathcal{D}, k = 1, \ldots, K \tag{3.5.32}$$

$$\boldsymbol{\rho}, \boldsymbol{\lambda} \succeq \boldsymbol{0}, \boldsymbol{\theta} \text{ unrestricted.} \tag{3.5.33}$$

Note that the variable $\boldsymbol{\rho}$ here is different from the previous one for INT-C, because of the existence of $\boldsymbol{\lambda}$ in the MP, as a result from the additional budget variable. At each iteration, we solve the following DP

$$\max \quad \sum_{a \in A} \Big[R_k(s,a) + \gamma \sum_{s' \in S} T_k(s'|s,a) \theta_{k,s'} - \sum_{i \in \mathcal{D}} d_i(s,a) \bar{\rho}_{k,i} \Big] \mu_{k,s,a} \tag{3.5.34}$$

$$\text{s.t.} \quad \sum_{a \in A} \mu_{k,s,a} = 1; \tag{3.5.35}$$

$$\mu_{k,s,a} \geq 0 \quad \forall\, a \in A. \tag{3.5.36}$$

Then we check the optimality condition using value

$$\nu_{k,s} = \sum_{a\in A} \bar{\mu}_{k,s,a}\Big[R_k(s,a) + \gamma \sum_{s'\in S} T_k(s'|s,a)\bar{\theta}_{k,s'} - \sum_{i\in\mathcal{D}} d_i(s,a)\bar{\rho}_{k,i}\Big]. \tag{3.5.37}$$

Optimality cuts to be added to the MP is formulated as

$$\theta_{k,s} \geq \sum_{a\in A} \bar{\mu}_{k,s,a}\Big[R_k(s,a) + \gamma \sum_{s'\in S} T_k(s'|s,a)\theta_{k,s'} - \sum_{i\in\mathcal{D}} d_i(s,a)\rho_{k,i}\Big] \quad \forall\ s \in S, k = 1,\ldots K. \tag{3.5.38}$$

Finally, constraints (3.5.12) in the integer model in Step-II is further modified:

$$\begin{aligned}\theta_{k,s} \leq \sum_{a\in A} \mu^*_{k,s,a}\Big[R_k(s,a) &+ \gamma \sum_{s'\in S} T_k(s'|s,a)\theta_{k,s'} \\ &- \sum_{i\in\mathcal{D}} d_i(s,a)\rho^*_{k,i}\Big] \quad \forall\ s \in S, k = 1,\ldots,K,\end{aligned} \tag{3.5.39}$$

where $\boldsymbol{\mu}^*$ and $\boldsymbol{\rho}^*$ are optimal values of variables from Step-I.

Chapter 4

Computational Analysis

In this section, we conduct computational analyses to evaluate the performances of the proposed algorithms. First, we conduct experiments on the MCLD algorithm and its variants in Chapter 3.1. Then, we conduct experiments on the TSBD method and its variants in Chapter 3.5.

We adopt four problems in the literature to test the performances of the algorithms, including a queueing problem (de Farias and Van Roy 2003), an inventory management problem (Puterman 2014; Lee et al. 2017), a machine maintenance problem (Puterman 2014), and a data transmission problem (Krishnamurthy 2016). The equipment replacement problem and the data transmission problem are modified from their original forms to allow arbitrary numbers of states and actions. For simplicity, in the following, we refer to the above testing problems as “queue”, “inventory”, “maintain” and “transmit”, respectively. Appendix A provides a detailed definition of four benchmarking problems.

4.1 Performance of The MCLD Algorithm

Although many fast MDP solution algorithms have been proposed in the literature, such as reinforcement learning (Sutton and Barto 2018), approximate dynamic programming (Warrington 2019; Braverman et al. 2020), or different MDP decomposition techniques (Kushner and Chen 1974; Abbad and Boustique 2003; Bertsimas and Mišić 2016), they either solve MDP approximately (Bertsimas and Mišić 2016; Warrington 2019; Braverman et al. 2020), or require special structures in MDP (Kushner and Chen 1974). Considering that the MCLD algorithm is developed as a generic way to solve MDP exactly regardless of its structure, we choose exact benchmark algorithms widely employed in many applications to solve MDP problems, e.g., the LP formulation of MDP, the dual of the LP formulation and the MPI algorithm (Puterman and Shin 1978).

According to the MCLD algorithm and its variants in Section 3.4, we conduct three experiments, on general MDP with no special properties, MDP with monotone optimal policy and CMDP. In addition, we also include randomly generate MDP instances for the MCLD algorithm, denoted by "random". Since the existence of the monotone optimal policy is not universal, three problems, "queue", "maintain" and "transmit", can be used as benchmarks for MDP with monotone optimal policy. All five problems are included for general MDP and CMDP. All experiments are conducted on a Linux server with 2.30GHz Intel Xeon Gold CPU and 256 GB memory. The LP models are solved with Gurobi via the Python interface. The MPI algorithm is implemented using the MDP Toolbox for Python library (Chadès et al. 2014).

4.1.1 General MDP

First, we compare the performance of the MCLD algorithm with the benchmark algorithms on general MDP problems without special structures. We consider all five problems discussed above, where each problem generates two sets of testing instances. Among each set, 10 testing instances are generated with the same number of states and actions. The testing problems and their configurations are shown in Table 4.1.

As we have discussed in Chapter 1, this study aims to derive an algorithm that solves MDP problems for which the LP formulation remains the sole solution method. Thus, we compare algorithm performances between the LP formulation, the dual of the LP and the MCLD algorithm in Table 4.2. We include two metrics to measure the CPU time, where t_{run} denotes the total run time of the algorithms, including the time for model construction and

Table 4.1: Configurations of benchmarking problems for general MDP.

Name	$\lvert S\rvert$	$\lvert A\rvert$	γ	Name	$\lvert S\rvert$	$\lvert A\rvert$	γ
random-1	100	100	0.999	inventory-2	502	501	0.999
random-2	500	500	0.999	maintain-1	100	101	0.999
queue-1	100	100	0.999	maintain-2	500	501	0.999
queue-2	500	500	0.999	transmit-1	110	101	0.999
inventory-1	102	101	0.999	transmit-2	520	501	0.999

Table 4.2: Performance comparison on general MDP problems.

Name	Best among LP and its dual		MCLD		Improvement	
	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(%)	t_{sol}(±CI)(%)
random-1	26.63 (± 0.19)	0.30 (±0.03)	4.30 (± 0.89)	0.04 (±0.01)	83.84 (±3.36)	88.25 (± 2.12)
random-2	3746.49 (±411.51)	66.27 (±6.85)	482.69 (±10.06)	3.34 (±0.14)	87.07 (±1.07)	94.94 (± 0.62)
queue-1	27.71 (± 0.42)	0.10 (±0.01)	7.72 (± 0.34)	0.04 (±0.01)	72.13 (±1.26)	60.85 (± 2.56)
queue-2	3308.82 (±406.16)	0.36 (±0.02)	719.75 (±20.09)	0.45 (±0.04)	78.15 (±1.80)	-25.36 (± 5.66)
inventory-1	26.88 (± 0.09)	0.11 (±0.03)	4.78 (± 1.35)	0.04 (±0.01)	82.20 (±5.06)	62.47 (± 19.87)
inventory-2	3578.38 (±506.20)	9.72 (±1.19)	421.74 (±90.82)	0.92 (±0.76)	88.12 (±3.04)	90.51 (± 7.76)
maintain-1	26.50 (± 0.21)	0.15 (±0.05)	2.83 (± 0.87)	0.02 (±0.01)	89.34 (±3.19)	86.85 (± 6.26)
maintain-2	3338.31 (±115.93)	20.19 (±1.66)	239.04 (± 6.74)	0.32 (±0.02)	**92.84** (±0.32)	**98.41** (± 0.20)
transmit-1	31.45 (± 0.40)	0.05 (±0.01)	7.17 (± 1.01)	0.04 (±0.01)	77.20 (±3.21)	23.25 (± 22.18)
transmit-2	3553.75 (±211.91)	2.96 (±0.38)	633.25 (±83.47)	0.54 (±0.05)	82.18 (±2.22)	81.63 (± 3.04)

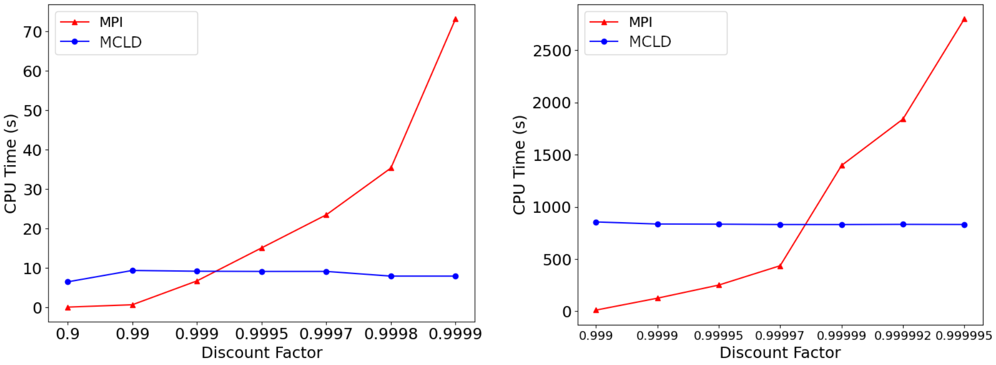
In the heading, "t_{run}" is the total run time of the algorithms, including both model construction and model solving; "t_{sol}" is the time taken to solve the model; "Improvement" shows the improvement of the MCLD algorithm against the best between the LP formulation and its dual. The metrics are shown in seconds (s).

the time for model solving, and t_{sol} includes only the time for Gurobi to solve the model. The table shows the average metrics over 10 instances for every problem, as well as the 95% confidence interval (CI). The performance improvements of the MCLD algorithm are calculated using the best among LP formulation and its dual as the benchmark.

In Table 4.2, all instances are solved to optimality. The MCLD algorithm outperforms the LP formulation of MDP and its dual in the total algorithm run time t_{run} for all problems. The results show over 75% improvements in t_{run} for most problems, and an up to 92.84% improvement for the maintain-2 problem. Moreover, the MCLD algorithm requires less model solving time t_{sol} for most problems, compared with the LP formulation and its dual. The improvements of t_{sol} ranges from 11.28% to 98.41%, depending on different problems. Note that for some problems, e.g., queue-2, the improvements of t_{sol} become negative for the MCLD algorithm, because it requires solving multiple linear programs iteratively until convergence, whereas conventional methods solve a single linear program with great efficiency. However, accounting for the time to build large-scale linear programs, improvements in the total algorithm run time t_{run} still show that the MCLD algorithm significantly outperforms the conventional methods.

4.1.2 Microscopic Analysis

Since the performance of the MPI algorithm depends heavily on the discount factor of the problem, we compare the performance of the MCLD algorithm with the MPI algorithm under different discount factors. Figure 4.1 shows the results of the comparison. We report the run time of the algorithms on two queue problems with 100 states, 100 actions and 500 states, 500 actions. Each problem generates 10 instances. The average run times are plotted for different discount factors. The figure suggests that the CPU time of the MPI algorithm increases exponentially when the discount factor becomes larger. For discount factors closer to 1, the MCLD algorithm shows a clear advantage over the MPI algorithm. Moreover, the performance of the MCLD algorithm remains stable across all discount factors.



(a) 100 states and 100 actions

(b) 500 states and 500 actions

Figure 4.1: Performance comparison between MPI and MCLD.

In order to show in detail how the MCLD algorithm optimizes MDP problems, we plot the values and policies for a queue problem with 20 states and binary actions ($A = \{0, 1\}$). Figure 4.2 shows the intermediate values calculated by the MCLD algorithm at each iteration. The optimal value is calculated by the LP formulation. Consistent with Corollary 3.1.3, by adding optimality cuts consecutively, the algorithm gradually improves the objective value of the MP, until it reaches optimality. In addition, in Figure 4.3, we plot the probabilities of taking action $a = 1$ as a representation of the intermediate policies learned by the MCLD algorithm. At iteration 0, immediately after initialization, the probabilities of taking action 1 are 0 for all states. Then, after several iterations, the MCLD algorithm gradually finds the optimal policy for each state. The policy improves over iterations and finally converges to the optimal policy at iteration 11.

In addition, we investigate the improvements of the MCLD algorithm over the LP formulation and the dual formulation under different combinations of states and actions. Specifically, we compare the algorithm performances using the random problem with 10, 20, 50, 100, 200, 500 states, and 10, 20, 50, 100, 200, 500 actions. We randomly generate 10 testing instances for each state-action combination. We plot the corresponding average improvements as heat maps in Figure 4.4, where darker colors suggest larger improvements. As the figure clearly indicates, the MCLD algorithm outperforms conventional methods more significantly when the problem scale becomes larger. Particularly, the figure shows improvements of up to 93% and 99% for t_{run} and t_{sol} respectively, on instances with 500 states and 500 actions. Note that for small-scale problems, e.g., with 10 states and 10 actions, the MCLD algorithm tends to spend more time (t_{sol}) solving linear programs, since the LP formulation or its dual only have to solve one program, and the MCLD algorithm solve linear programs iteratively until convergence.

4.1.3 Comparing With VI

As shown in Section 4.1.1, the MCLD algorithm operates in a similar way as VI. Both algorithms take multiple iterations to improve the value of the each state. Although the

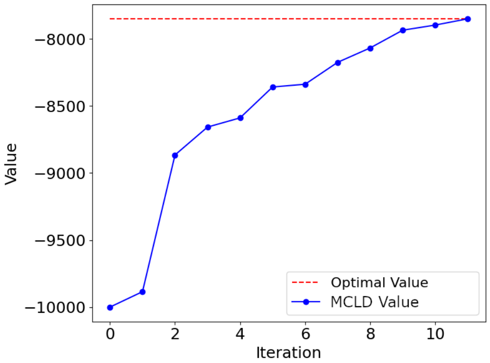


Figure 4.2: The convergence of values generated by MCLD.

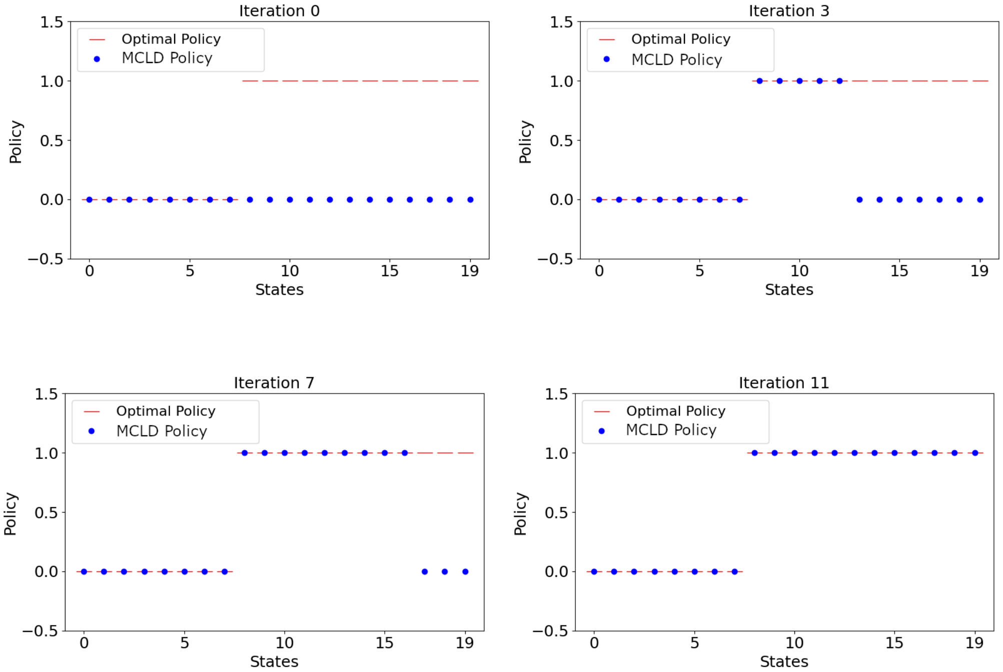
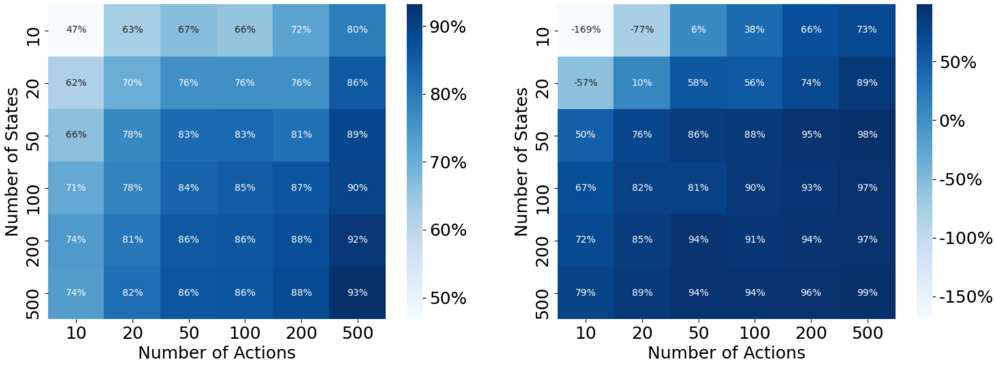


Figure 4.3: The convergence of policies generated by MCLD.



(a) t_{run}

(b) t_{sol}

Figure 4.4: Heat maps of MCLD improvements.

value updates in Algorithm 1 resembles the value updates in VI, the MCLD algorithm still differs from VI in a significant way. In VI, the values are updated using the equation

$$v_s = \max_{a \in A} \Big\{ R(s,a) + \gamma \sum_{s' \in S} T(s', s, a) v_{s'} \Big\}, \tag{4.1.1}$$

where v_s is the value of state s. Thus, $v_{s'}$ is a fixed number for each possible future state s'. However, in the MCLD algorithm, the values θ_s are updated using the cut

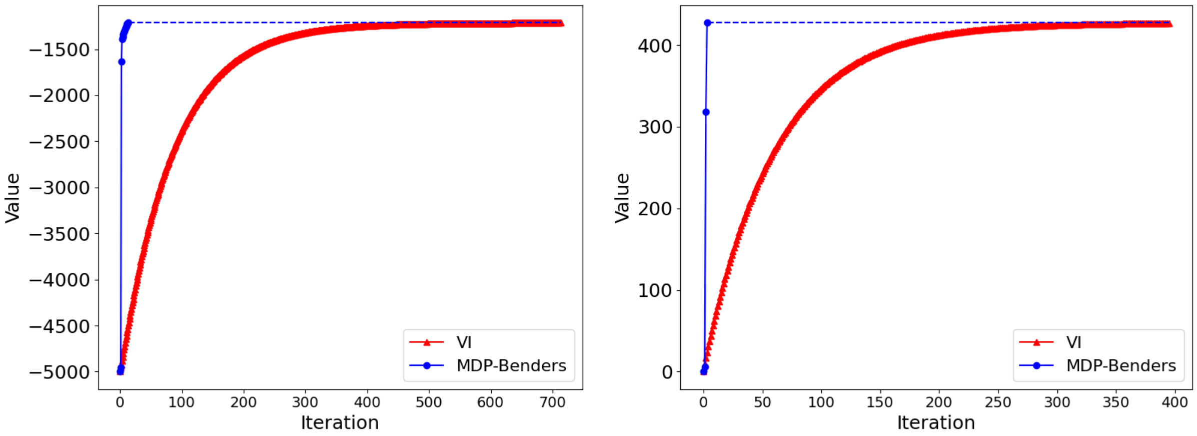
$$\theta_s \geq \sum_{a \in A} \bar{y}_{s,a} \Big[R(s,a) + \gamma \sum_{s' \in S} T(s'|s,a) \theta_{s'} \Big], \quad \forall\ s \in S, \tag{4.1.2}$$

where θ_s and $\theta_{s'}$ are both variables whose values may change. In this case, even though the objective of the MP is to minimize the value of θ_s, it is not trivially updated as Equation (4.1.1), but in a more complex way, calculated as a multi-dimensional polyhedron by a linear program.

Figure 4.5 shows experiment results from comparing MCLD with VI. The algorithms start at the same values for all states. The convergence threshold for VI is set to be 0.01. The figure shows significant differences in algorithm behaviors. The VI algorithm improves the value gradually, by small increments, resulting in a longer convergence time. In contrast, the MCLD algorithm converges much faster, with large leaps between iterations. Thus, by solving linear programs, the MCLD algorithm is able to achieve larger improvements at each value update, leading to a faster convergence rate than VI.

4.1.4 Special MDP

In this section, we test the performance of the MCLD algorithm on some of the special MDP problems introduced in Section 3.4. First, we consider MDPs with monotone optimal policies. Since the existence of the monotone optimal policy is guaranteed by the sufficient conditions, as introduced in Section 3.4.2, we choose three classes of testing problems, queue, maintain and transmit, for which the existence of monotone optimal policies have been proved (Puterman 2014; Krishnamurthy 2016). In total, we include six problems as shown in Table 4.3. For each problem, 10 testing instances are randomly generated.



(a) A queue instance

(b) A inventory instance

Figure 4.5: Comparing the convergence between MCLD and VI.

Table 4.3: Configurations of testing problems with monotone optimal policies.

Name	$\|S\|$	$\|A\|$	γ
queue-1	100	100	0.999
queue-2	500	500	0.999
maintain-1	100	101	0.999
maintain-2	500	501	0.999
transmit-3	2000	2	0.999
transmit-4	4000	2	0.999

In Section 4.1.1, we have shown the superior performance of the MCLD algorithm compared with conventional methods as such the LP formulation and its dual. Thus, here, we use the MCLD algorithm as the benchmark and compare the performance with its extension, i.e., the MCLD algorithm adding cuts (3.4.16) specifically designed for MDP with monotone optimal policy. In the following, we refer to the MCLD algorithm that adds cuts (3.4.16) as the MCLD algorithm with monotone optimal policy (MCLD-MOP). Similar to previous experiments, we focus on two metrics, t_{run} and t_{sol}, representing the CUP time to run the entire algorithm, and the CPU time to solve linear programs by Gurobi.

Table 4.4 shows the results of the comparison. All instances are solved to optimality. Overall, MCLD-MOP solves the testing problems 50.40%–88.77% faster than the general MCLD algorithm. By adding cuts (3.4.16), MCLD-MOP is able to prune suboptimal actions thus reducing the number of variables in the linear programs. As a result, MCLD-MOP spends up to 94.56% less time in solving linear programs.

Next, we conduct experiments on CMDP problems. Due to the additional constraints of CMDP, we reduce the testing problem size so that the run time of larger instances remains tractable. The problem configurations are shown in Table 4.5. For all CMDP problems, the costs of additional constraints, i.e., $d_i(s, a)$ and D_i, for all $i \in \mathcal{D}$, are randomly generated. In addition, we let $|\mathcal{D}| = |S|$, meaning that the number of additional constraints matches the number of states. Similar to the above experiments, we generate 10 instances for each problem and collect the average metrics with 95% CI. In the following, we use MCLD-CMDP to refer to the MCLD algorithm with cuts (3.4.31).

Results of the algorithm performances are summarized in Table 4.6. All instances are solved to optimality. Consistent with previous results, by decomposing the MDP, the MCLD-CMDP algorithm shows significant advantages over conventional methods. Specifically, the MCLD-CMDP algorithm saves the total run time t_{run} and the model solving time t_{sol} by up to 92.06% and 99.38%, respectively. Note that although the MCLD-CMDP algorithm solves several problems with large CI for t_{run}, such as queue-3, the improvements in t_{run} consistently show smaller CI, indicating that the variations in t_{run} are caused by the differences in the

Table 4.4: Performance comparison on problems with monotone optimal policies.

Name	MCLD		MCLD-MOP		Improvement	
	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(%)	t_{sol}(±CI)(%)
queue-1	9.09 (± 0.05)	0.05 (±0.01)	1.18 (±0.01)	0.01 (±0.01)	86.98 (± 0.10)	77.40 (± 5.16)
queue-2	881.45 (± 9.81)	0.60 (±0.05)	123.89 (±1.76)	0.17 (±0.03)	85.94 (± 0.18)	72.11 (± 3.95)
maintain-1	2.91 (± 0.90)	0.02 (±0.01)	1.21 (±0.01)	0.01 (±0.01)	57.10 (±10.19)	47.48 (±32.06)
maintain-2	257.03 (± 1.96)	0.42 (±0.02)	127.48 (±1.00)	0.14 (±0.02)	50.40 (± 0.23)	67.32 (± 4.06)
transmit-3	66.84 (± 18.72)	0.33 (±0.16)	7.20 (±0.03)	0.08 (±0.01)	**88.77** (± 3.80)	72.13 (±17.76)
transmit-4	1336.57 (±115.84)	8.45 (±1.37)	174.00 (±0.80)	0.45 (±0.04)	86.93 (± 1.29)	**94.56** (± 1.12)

In the heading, "t_{run}" is the total run time of the algorithms, including both model construction and model solving; "t_{sol}" is the time taken to solve the model; "Improvement" shows the improvement of MCLD-MOP against the general MCLD algorithm. The metrics are shown in seconds (s).

Table 4.5: Configurations of testing problems for CMDP.

Name	$\lvert S\rvert$	$\lvert A\rvert$	γ	Name	$\lvert S\rvert$	$\lvert A\rvert$	γ
random-1	100	100	0.999	inventory-3	402	401	0.999
random-3	400	400	0.999	maintain-1	100	101	0.999
queue-1	100	100	0.999	maintain-3	400	401	0.999
queue-3	400	400	0.999	transmit-1	110	101	0.999
inventory-1	102	101	0.999	transmit-5	420	401	0.999

Table 4.6: Performance comparison on CMDP problems.

Name	Best among LP and its dual		MCLD-CMDP		Improvement	
	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(%)	t_{sol}(±CI)(%)
random-1	54.10 (± 0.76)	0.62 (±0.10)	8.79 (± 2.80)	0.04 (±0.01)	83.75 (±5.22)	93.01 (±2.48)
random-3	3661.83 (± 52.88)	83.82 (±3.50)	563.47 (± 14.69)	1.85 (±0.07)	84.61 (±0.50)	97.79 (±0.14)
queue-1	51.65 (± 0.62)	0.24 (±0.01)	16.80 (± 0.05)	0.06 (±0.00)	67.46 (±0.41)	76.18 (±0.72)
queue-3	3439.00 (±122.72)	33.97 (±4.70)	1034.41 (±161.26)	0.72 (±0.09)	69.89 (±4.98)	97.86 (±0.38)
inventory-1	56.46 (± 0.40)	0.34 (±0.04)	8.78 (± 1.93)	0.04 (±0.01)	84.46 (±3.41)	87.69 (±3.65)
inventory-3	3493.91 (± 79.20)	29.98 (±2.41)	466.35 (±102.13)	0.62 (±0.36)	86.64 (±3.03)	97.94 (±1.10)
maintain-1	52.76 (± 0.54)	0.41 (±0.04)	6.03 (± 1.88)	0.02 (±0.01)	88.58 (±3.53)	94.27 (±2.74)
maintain-3	3612.09 (± 65.55)	33.30 (±2.97)	286.76 (± 37.15)	0.21 (±0.02)	**92.06** (±1.03)	**99.38** (±0.08)
transmit-1	64.51 (± 0.46)	0.39 (±0.04)	14.64 (± 0.55)	0.05 (±0.00)	77.31 (±0.95)	87.78 (±1.85)
transmit-5	3897.06 (±165.33)	32.46 (±9.11)	790.12 (±113.69)	0.66 (±0.08)	79.69 (±3.38)	97.93 (±0.39)

In the heading, "t_{run}" is the total run time of the algorithms, including both model construction and model solving; "t_{sol}" is the time taken to solve the model; "Improvement" shows the improvement of MCLD-CMDP against the best between the LP formulation and its dual. The metrics are shown in seconds (s).

random instances, and the improvements of the MCLD-CMDP algorithm remain stable across different instances for the same problem.

4.2 Performance of The TSBD Method

In this section, we evaluate the algorithm performances for the LSSD framework. Since there is no available algorithm in the literature that solves the LSSD framework exactly, we compare the performances of the NLP formulation, the alternate INT formulation, and the TSBD method.

Here, we utilize two benchmarking problems from previous experiments, namely queue and inventory, for which the extension to include strategic decisions comes naturally. For queue, the strategic decision is to decide on the optimal maximum queue length, before making operational decisions about service rates. For inventory, the strategic decision is to choose the optimal inventory capacity, and the operational decision is to choose the order timing and quantity to fill in the inventory. Detailed extension and formulation of the benchmarking problems are presented in Appendix B.

For each benchmarking problem, we include three configurations, which can be discretized into 10, 50, and 100 MDP models, respectively, representing small, medium, and large-scale problems. Table 4.7 summarizes the configurations for al benchmarking problems. In addition, 10 testing instances are generated with randomized parameters to avoid outliers. All experiments are conducted on a Linux server with 2.30GHz Intel Xeon Gold CPU and 256 GB memory. The linear and nonlinear models are solved with Gurobi via the Python interface. The average value over 10 instances and the 95% CI are reported in the results.

Table 4.8 shows the results of comparing NLP, INT, and TSBD for the LSSD framework on general MDP problems. The table is arranged in such a way that instance sizes increase from top to bottom. In general, the discretized INT formulation solves LSSD faster than NLP, and the TSBD algorithm outperforms the NLP and INT formulation, with overall improvements of up to over 80% in the total algorithm runtime (t_{run}), and up to over 91% in the time to solve LP models (t_{sol}). Importantly, the improvements increase as the instance

Table 4.7: Configurations of testing problems for the framework with regular MDP.

Name	K	$\lvert S\rvert$	$\lvert A\rvert$	γ	Name	K	$\lvert S\rvert$	$\lvert A\rvert$	γ
queue-4	10	10	10	0.999	inventory-4	10	12	11	0.999
queue-5	50	50	50	0.999	inventory-5	50	52	51	0.999
queue-6	100	100	100	0.999	inventory-6	100	102	101	0.999

In the table, $|S|$ and $|A|$ represents the number of states and actions of one MDP model.

Table 4.8: Performance comparison between NLP, INT, and TSBD.

Name	NLP		INT		TSBD		Improvement	
	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(%)	t_{sol}(±CI)(%)
queue-4	0.21 (±0.03)	0.13 (±0.03)	0.02 (±0.00)	0.02 (±0.00)	0.07 (±0.00)	0.02 (±0.00)	-193.48 (±35.19)	-3.45 (±20.78)
inventory-4	3.67 (±1.52)	2.91 (±1.51)	0.02 (±0.00)	0.02 (±0.00)	0.04 (±0.00)	0.01 (±0.00)	-76.53 (±19.87)	43.76 (±5.74)
inventory-5	1179.10 (±325.11)	889.69 (±48.53)	7.54 (±1.31)	4.02 (±1.06)	3.35 (±0.29)	0.77 (±0.16)	54.99 (±8.25)	80.62 (±3.11)
queue-5	214.02 (±24.29)	193.16 (±23.76)	8.08 (±1.14)	2.72 (±0.95)	6.70 (±0.53)	0.96 (±0.06)	16.33 (±12.65)	61.45 (±21.64)
queue-6	3915.13 (±8.02)	3600.00[†] (±0.00)	228.63 (±70.38)	98.23 (±55.77)	58.16 (±1.05)	8.60 (±0.16)	72.92 (±12.37)	87.54 (±15.28)
inventory-6	4075.34 (±251.71)	3600.00[†] (±0.00)	192.40 (±39.50)	119.01 (±41.98)	37.64 (±7.31)	10.45 (±3.07)	**80.38** (±1.51)	**91.10** (±1.51)

[†]: The optimal solutions to some instances are not found within the time limit.

size grows, showing that the proposed TSBD method is more specialized in solving larger instances than the NLP and INT formulation. Note that for smaller instances, the INT model has the computational advantage since it avoids nonlinear constraints in NLP and multiple SPs in TSBD, but the excessive integer variables make it less efficient for larger instances.

Table 4.9 shows the results of comparing NLP-C, INT-C, and TSBD-C for the LSSD framework on CMDP problems. The instances are arranged with increased sizes from top to bottom. In this case, due to the extra linear constraints, all models become more difficult to solve than previous experiment. As a result, even in small instances, the TSBD-C algorithm shows advantages compared with NLP-C and INT-C. The NLP formulation shows the worse performance, and cannot solve medium instances with around 50 states and actions in the MDP. The INT-C formulation still outperforms NLP-C because of the reduced nonlinear constraints, but it is not able to solve a few of the larger instances. Compared with the conventional methods, TSBD-C not only solves all instances to the true optimum, but also does so in an efficient way, with up to over 78% improvements in t_{run}, and up to over 96% improvements in t_{sol}.

Table 4.10 shows the results of comparing NLP-VB, INT-VB, and TSBD-VB for the LSSD framework on CMDP problems with variable budgets. The instances are arranged with increased sizes from top to bottom. Similar to CMDP, improvements of the TSBD-VB algorithm can be observed in small instances due to the complexity of the problem. The overall performance of the TSBD-VB algorithm improves by up to 76% in t_{run}, and over 93% in t_{sol}, compared with those of NLP-VB and INT-VB.

Table 4.9: Performance comparison between NLP-C, INT-C, and TSBD-C.

Name	NLP-C		INT-C		TSBD-C		Improvement	
	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(%)	t_{sol}(±CI)(%)
queue-4	0.42 (±0.09)	0.34 (±0.08)	0.44 (±0.07)	0.07 (±0.02)	0.36 (±0.04)	0.05 (±0.01)	11.23 (±21.27)	32.01 (±24.21)
inventory-4	35.85 (±29.64)	35.42 (±29.65)	0.41 (±0.02)	0.05 (±0.01)	0.26 (±0.03)	0.02 (±0.00)	36.16 (±8.61)	50.32 (±10.68)
queue-5	237.76 (±134.09)	198.76 (±134.29)	239.52 (±53.35)	46.08 (±31.71)	67.76 (±2.00)	5.79 (±0.34)	61.76 (±21.40)	85.77 (±3.79)
inventory-5	3696.23 (±82.36)	3600.00† (±0.00)	291.17 (±56.20)	98.00 (±50.19)	60.66 (±0.95)	4.32 (±0.41)	**78.85** (±3.79)	95.24 (±1.68)
queue-6	3664.49 (±8.36)	3600.00† (±0.00)	4967.31 (±846.24)	1430.72 (±539.15)	970.93 (±27.21)	97.01 (±5.09)	73.50 (±0.01)	92.78 (±0.03)
inventory-6	4168.08 (±11.48)	3600.00† (±4.70)	6678.86 (±167.05)	3563.57† (±154.93)	976.45 (±47.67)	109.01 (±26.32)	76.57 (±0.01)	**96.94** (±0.01)

†: The optimal solutions to some instances are not found within the time limit.

Table 4.10: Performance comparison between NLP-VB, INT-VB, and TSBD-VB.

Name	NLP-VB		INT-VB		TSBD-VB		Improvement	
	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(s)	t_{sol}(±CI)(s)	t_{run}(±CI)(%)	t_{sol}(±CI)(%)
queue-4	0.43 (±0.05)	0.35 (±0.05)	0.41 (±0.03)	0.09 (±0.04)	0.30 (±0.02)	0.05 (±0.00)	25.38 (±5.25)	43.65 (±16.76)
inventory-4	38.62 (±23.08)	38.16 (±23.07)	0.50 (±0.07)	0.10 (±0.05)	0.26 (±0.02)	0.03 (±0.00)	46.56 (±9.67)	72.71 (±12.36)
queue-5	154.45 (±49.25)	147.52 (±49.26)	251.05 (±26.79)	62.90 (±25.47)	72.80 (±2.46)	12.75 (±1.30)	50.46 (±17.61)	78.76 (±5.49)
inventory-5	3132.69 (±1648.45)	3078.08 (±1648.27)	275.68 (±29.62)	89.10 (±29.13)	65.89 (±2.62)	9.13 (±1.67)	**76.00** (±2.24)	89.32 (±3.46)
queue-6	3662.91 (±8.94)	3600.00† (±0.17)	6642.40 (±1466.72)	2922.98 (±1297.26)	1221.49 (±55.64)	348.78 (±33.71)	66.65 (±1.55)	86.58 (±7.77)
inventory-6	4200.15 (±101.86)	3600.00† (±0.00)	6674.53 (±603.93)	3539.62† (±276.00)	1140.66 (±259.58)	242.58 (±83.40)	72.87 (±5.81)	**93.13** (±2.20)

†: The optimal solutions to some instances are not found within the time limit.

Chapter 5

Defending Interdependent CIS

In this chapter, we apply the LSSD framework to a real-world critical infrastructure protection problem. Critical infrastructure systems (CISs) are major arteries of modern society. Economic prosperity, social welfare, and public security all heavily depend on CISs. According to the U.S. Department of Homeland Security (DHS), CISs are “vital physical and cyber systems whose incapacity or destruction would have a debilitating impact” on national security. DHS has characterized 16 CIS sectors, including but not limited to energy, water, transportation, commercial facility, communication, food and agriculture, healthcare, etc (The Cybersecurity and Infrastructure Security Agency 2021).

Often, CISs are not isolated, but highly interconnected and interdependent (Ouyang 2014). For example, water systems support the daily operations of the healthcare systems by providing safe and clean water. Commercial and financial sectors rely on information technologies to secure transactions. Nearly all systems require electricity generated by the power grid or nuclear facilities. The incapability of any CIS not only causes a shortage of service from that particular system, but also reduces service qualities of other interconnected systems. Thus, a cascading effect could appear following the failure of one CIS.

The interconnectivity of CIS has made them vulnerable when facing attacks. In the last two decades, the disastrous consequences of CIS cascading failures have been tested by incidents occurring all over the globe, such as the 2001 World Trade Center attack, the 2003 Northeast blackout, the 2016 Brussels bombings, and the 2017 hurricane Harvey. For instance, according to the U.S. Department of Energy, the 2003 blackout inflicted 6 billion dollars worth of damage, with collateral impacts on water supply, transportation, communication, and hospitals (U.S. Department of Energy 2021).

Facing potential attacks from natural disasters or terrorist groups, governments and international organizations are facilitating legislation to protect CIS facilities. In 2006, the European Union (EU) has launched the European Programme for Critical Infrastructure Protection (EPCIP), with the aim to reinforce and protect CIS facilities in all EU nations. Likewise, the U.S. government has issued Presidential Policy Directive 21 (PPD-21), which makes CIS security a national policy to ensure the resilience of CIS sectors. China has also legislated CIS protection, especially in the areas of internet and information infrastructures.

5.1 Current Literature

In the literature, several review papers have been published, summarizing hundreds of studies that analyze CIS resilience from various perspectives (Yusta et al. 2011; Ouyang 2014). Recent mainstream analytical approaches of CIS resilience can be categorized into six types (Ouyang 2014): data-focused empirical analysis, agent-based simulation, system dynamics, economic perspective, network-based method, and others. Especially, the network-based method is one of the most widely adopted approaches due to its capability of modeling physical connections and commodity flows between CISs (Ouyang 2017; Ghorbani-Renani et al. 2020; Galbusera et al. 2020). The network-based method models the CIS topology with networks (graphs), which use nodes to represent individual infrastructure facilities and edges to represent the connections or transmissions between facilities. With already-established theoretical results, the network models of CISs usually produce mathematical properties that are helpful in deriving solution algorithms (Ouyang and Fang 2017; Fang and Zio 2019). Network-based methods can also be easily applied when game theory is involved (Brown et al. 2006; Baykal-Güersoy et al. 2014; Ferdowsi et al. 2017). In that case, a rational attacker is introduced to plan attacks against the CIS so that maximum damage is inflicted. Then, the defender makes decisions to best protect the CIS or to optimally restore its service.

Despite its wide applications, network-based models often suffer from high computational cost (Ma et al. 2013b). As a network grows larger, the scale of the model grows exponentially. Thus, commonly, network-based models only consider two-step (attack–defend) or three-step (defend–attack–defend) problems (Ouyang 2017; Brown et al. 2006). However, in the real world, attacks against the CISs often come in batches. For example, in 2015, six coordinated attacks occurred in Paris within four hours, targeting stadiums, restaurants, and theaters. Similarly, in 2019, seven populated areas (churches and hotels) in Sri Lanka were attacked within a six-hour window. These coordinated attacks demand resources be re-distributed repeatedly between attack intervals, so that CISs can be best protected.

In the current literature, only a few have considered sequential attacks under the context of CISs protection (Jones et al. 2006; Ma et al. 2013a), but many have proposed innovative models from game theory perspectives (Kaplan et al. 2010; Zhuang et al. 2010; Hausken and Zhuang 2011; Shan and Zhuang 2013; Jose and Zhuang 2013; Chang et al. 2015; Rass and Zhu 2016; Shan and Zhuang 2018). Especially, Markov game is one of the widely used modeling approaches (Zhuang et al. 2010; Ma et al. 2013a; Chang et al. 2015), due to its capability of modeling long-term interactions between the attacker and the defender.

However, many of the above models are based on assumptions that are not realistic enough. Specifically, in many studies, defenders are able to assign defense resources at each period in multi-period games (Zhuang et al. 2010; Hausken and Zhuang 2011; Shan and Zhuang 2018), neglecting the cost of preparing the defense resources in advance, or the cost of constructing infrastructures such as warehouses or operation bases that support resource distribution. Although budget constraints are applied to the allocation of defense resources (Hausken and Zhuang 2011; Shan and Zhuang 2018), the budgets are often fixed parameters chosen prior to the defenders' moves. In consequence, as one of the important aspects of the model, the resources available for a defender to use against the attacks, are not optimized. Moreover, to model with game theory, it is assumed that the attacker, who causes infrastructural failure, either myopic or not, is at least rational (Zhuang et al. 2010;

Hausken and Zhuang 2011; Chang et al. 2015; Shan and Zhuang 2018), whereas in the real world, failures come from both terrorist attacks and natural disasters, among which the later cannot be rationalized.

Thus, in this study, we consider a CIS protection problem from the defender's view. We assume that the defender only has partial, stochastic information on the attacker's intention. The attacker engages the targeted CIS facilities in a sequential manner, with an unknown number of attacks. Using the LSSD framework from previous chapters, we propose an optimization method that jointly makes network design and resource allocation decisions in strategic planning, and devises defense policies in response to each of the attacks with limited resources.

Figure 5.1 provides a demonstration of the problem considered. Initially, the defender makes strategic decisions to establish interconnectivity between facilities, and allocate defense resources to every facility. When the attacks occur, the defender chooses defense strategies according to the current situation and protects all facilities within the CIS network, in order to best protect the facilities from dysfunction. To model the problem, we adopt the LSSD framework established in previous chapters, where the first stage makes strategic decisions on the CIS network design, and the second stage makes sequential operational decisions at attack intervals on the defense strategies.

5.2 Formulation of the CIS Model

We use a graph $G := (V, E)$ to model the CIS network. The set of nodes V considers two types of facilities, i.e., $V := V^I \cup V^D$, where V^I denotes independent CIS facilities, and V^D denotes dependent CIS facilities. We assume that the facility $i \in V^D$ must be connected to another facility $j \in V^I$ to generate output. The edges E denotes possible connections between CIS facilities. The connection can be further modeled with an adjacency matrix $\mathcal{A}$, where $\mathcal{A}_{i,j} = 1$ if $(i, j) \in E$.

As discussed, the first stage makes two types of decisions, network design, and resource allocation. We use $u_{i,j} \in \{0, 1\}$, $\forall\, i \in V^D$, $j \in V^I$ to denote the service between independent

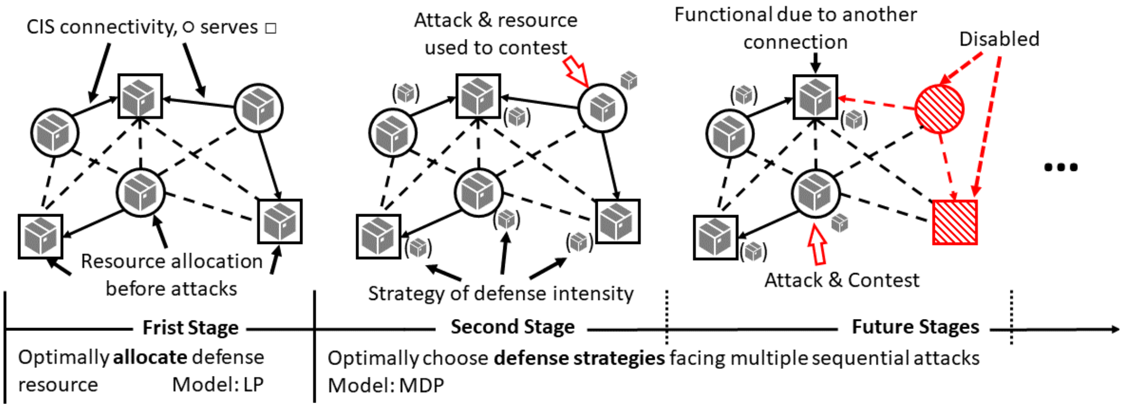

CIS connectivity, ○ serves □
Resource allocation before attacks
Attack & resource used to contest
Strategy of defense intensity
Functional due to another connection
Disabled
Attack & Contest
Frist Stage
Second Stage
Future Stages
Optimally allocate defense resource
Model: LP
Optimally choose defense strategies facing multiple sequential attacks
Model: MDP

Figure 5.1: An demonstration of the CIS defense problem.

and dependent systems. When $u_{i,j} = 1$, facility at location j serves the facility at location i. We use x_i, $\forall\ i \in V$ to denote the defense resource allocated to the location i. The allocated defense resource x_i represents the maximum amount of defense resources that can be consumed when planning the defense strategies in the second stage. In addition, we let $c^s_{i,j}$, $i \in V^D$, $j \in V^I$ be the cost of establishing service between i and j, and c^r the cost of unit defense resource. The first stage model can be written as the following integer program.

$$\max \quad -\sum_{i\in V} c^r x_i - \sum_{j\in V^I}\sum_{i\in V^D} c^s_{i,j} u_{i,j} \tag{5.2.1}$$

$$\text{s.t.} \quad \sum_{j\in V^I} u_{i,j} \geq 1 \quad \forall\ i \in V^D; \tag{5.2.2}$$

$$u_{i,j} \leq \mathcal{A}_{i,j} \quad \forall\ i \in V^D, j \in V^I; \tag{5.2.3}$$

$$x_i \in \mathbb{N}_+ \quad \forall\ i \in V; \tag{5.2.4}$$

$$u_{i,j} \in \{0,1\} \quad \forall\ i \in V^D, j \in V^I. \tag{5.2.5}$$

The second stage is modeled using a discounted, infinite-horizon MDP. The infinite-horizon represents the lack of information about the attacker's intentions, including the number of attacks. Modeling with infinite-horizon also allows the model to produce stationary policies, i.e., no matter the number of attacks, the optimal strategy would remain the same. As such, the decision making epochs are defined as $t = 1, 2, \ldots, \infty$.

States of the MDP represent the status of the all CIS facilities in the network. We let $s := (s_1, s_2, \ldots, s_{|V|}) \in S$, where the facility at $i \in V$ is functional if $s_i = 1$, and disabled if $s_i = 0$. For example, $s = (0, 1, 1)$ denotes functional CIS facilities at $i = 2$ and $i = 3$, and disabled facility at $i = 1$.

Actions $a := (a_1, a_2, \ldots, a_{|V|}) \in A$ represent the defense intensity at each $i \in V$. We let $a_i \in \{0, 1, 2, \ldots, M\}$, $\forall\ i \in V$, suggesting that a_i amount of defense resources will be consumed if location i is attacked. For example, $a = (0, 2, 3)$ denotes that if $i = 1$, $i = 2$, or $i = 3$ is attacked, 0, 2, or 3 defense resources will be consumed, respectively.

The transition probability accounts for the probability of attack, as well as the contest between the defender and the attacker. We assume that the defender knows the attacker's intension up to a probability distribution, where $P_s(i) \in [0, 1]$ denotes the probability that

$i \in V$ is attacked under state s. We let $P_s(i) = 0$ for all $i \in V, s_i = 0$, suggesting that the defender do not consider attacks on disabled facilities. When all facilities are disabled, i.e., $s = (0, 0, \ldots, 0)$, we let $P_s(i) = P_s(j)$, $\forall\ i, j \in V$, since at this state, none of the defense strategies would increase the defender's reward. We also consider the possible attack intensities $\beta = 1, 2, \ldots, M$ at locations $i \in V$, and use $P_i(\beta) \in [0, 1]$ to denote the probability that intensity β is chosen at location i. Then, the probability of attacking i with intensity β at state s, $P_s(i, \beta)$, can be calculated as

$$P_s(i, \beta) = P_s(i) \cdot P_i(\beta). \tag{5.2.6}$$

To model the contest between the attacker and the defender with resources, we use the following contest function, producing the success probability of an attack (Tullock 2001; Skaperdas 1996):

$$P(\text{success}) = \frac{\beta}{\beta + a}, \tag{5.2.7}$$

where β and a denote attack and defense intensities, respectively. At a state s, the probability of attacking i and succeeding is

$$P_s(i) \cdot \sum_{\beta=1}^{M} P_i(\beta) \cdot \frac{\beta}{\beta + a}. \tag{5.2.8}$$

The transition probabilities can be represented as follows.

$$T(s'|s, a) := \begin{cases} 1, & \text{if } s' = s = (0, 0, \ldots, 0); \\ \sum_{i \in V} P_s(i) \cdot \sum_{\beta=1}^{M} P_i(\beta) \cdot \frac{a_i}{\beta + a_i}, & \text{if } s \neq (0, 0, \ldots, 0), s' = s; \\ P_s(i) \cdot \sum_{\beta=1}^{M} P_i(\beta) \cdot \frac{\beta}{\beta + a_i}, & \text{if } \exists\ i \in V, s'_i = 0, s_j = 1, s'_j = s_j, \forall\ j \in V, j \neq i \\ 0, & \text{otherwise.} \end{cases} \tag{5.2.9}$$

The reward of the MDP accounts for the output of different CIS facilities. We use $r_i^I \in \mathbb{R}_+$, $\forall\ i \in V^I$ to denote the outputs from independent CIS facilities and $r_i^D \in \mathbb{R}_+$, $\forall\ i \in V^D$ the outputs from dependent CIS facilities. We let $\delta_i \in [0, 1]$ be the proportion of output for a CIS facility $i \in V^D$ that is dependent on other facilities in V^I. We further define a binary variable $\sigma_{s,i}$, $\forall\ s \in S, i \in V^D$ to represent that under state s, whether facility i is connected to at least one functional facility that provides service. Thus, the reward $R(s,a)$ can be calculated using the following constraints:

$$\begin{aligned}
R(s,a) &= \sum_{i\in V^I} s_i \cdot r_i^I + \sum_{i\in V^D} \Big[(1-\delta_i)\cdot s_i \cdot r_i^D + \delta_i \cdot \sigma_{s,i} \cdot s_i \cdot r_i^D\Big] \\
&= \sum_{i\in V^I} s_i \cdot r_i^I + \sum_{i\in V^D} \Big[1 + (\sigma_{s,i}-1)\cdot \delta_i\Big] \cdot s_i \cdot r_i^D \quad \forall\ s \in S, a \in A; \qquad (5.2.10)\\
M \cdot \sigma_{s,i} &\geq \sum_{j\in V^I} s_j \cdot u_{i,j} \quad \forall\ s \in S, i \in V^D; \qquad (5.2.11)\\
\sigma_{s,i} &\leq \sum_{j\in V^I} s_j \cdot u_{i,j} \quad \forall\ s \in S, i \in V^D, \qquad (5.2.12)
\end{aligned}$$

where M is a large number, such as $M = |V^I|$.

The capacity of defense resources is limited by the strategic decisions made at the first stage. To model the scarcity of resources, we add additional linear constraints to the MDP. The decision variable $y_{s,a}$ for the second stage represents the number of times that the defense plan a is executed under CIS state s, and a_i represents the resource consumed at i under the defense strategy a. Thus, we use the following linear constraints to ensure that the expected defense resources consumed at $i \in V$ are within the initially allocated capacity.

$$\sum_{s\in S}\sum_{a\in A} a_i \cdot y_{s,a} \leq x_i \quad \forall\ i \in V. \qquad (5.2.13)$$

Then, the second stage CMDP model can be formulated as follows.

$$\begin{aligned}
\max \quad & \sum_{s\in S}\sum_{a\in A} R(s,a) y_{s,a} && (5.2.14)\\
\text{s.t.} \quad & \sum_{a\in A} y_{s,a} - \gamma \sum_{s'\in S}\sum_{a\in A} T(s|s',a) y_{s',a} = \alpha(s) \quad \forall\ s \in S; && (5.2.15)\\
& \sum_{s\in S}\sum_{a\in A} a_i \cdot y_{s,a} \leq x_i \quad \forall\ i \in V; && (5.2.16)\\
& y_{s,a} \geq 0 \quad \forall\ s \in S, a \in A. && (5.2.17)
\end{aligned}$$

Since the transition probability in the model is not dependent on first-stage decisions, the variables $\boldsymbol{x}$ and $\boldsymbol{z}$ only lead to different reward structures. Thus, we use the variable $r_{s,a}$ to denote the reward under state s and action a, subject to the influence of first-stage strategic decisions. The NLP formulation of the two-stage model is shown as follows.

$$\max \quad -\sum_{i\in V} c^r x_i - \sum_{j\in V^I}\sum_{i\in V^D} c^s_{i,j} u_{i,j} + \sum_{s\in S}\sum_{a\in A} R_{s,a} y_{s,a} \tag{5.2.18}$$

$$\text{s.t.} \quad \sum_{j\in V^I} u_{i,j} \geq 1 \quad \forall\, i \in V^D; \tag{5.2.19}$$

$$u_{i,j} \leq \mathcal{A}_{i,j} \quad \forall\, i \in V^D, j \in V^I; \tag{5.2.20}$$

$$R(s,a) = \sum_{i\in V^I} s_i \cdot r^I_i + \sum_{i\in V^D} \Big[1 + (\sigma_{s,i} - 1)\cdot \delta_i\Big] \cdot s_i \cdot r^D_i \quad \forall\, s \in S, a \in A; \tag{5.2.21}$$

$$M \cdot \sigma_{s,i} \geq \sum_{j\in V^I} s_j \cdot u_{i,j} \quad \forall\, s \in S, i \in V^D; \tag{5.2.22}$$

$$\sigma_{s,i} \leq \sum_{j\in V^I} s_j \cdot u_{i,j} \quad \forall\, s \in S, i \in V^D; \tag{5.2.23}$$

$$\sum_{a\in A} y_{s,a} - \gamma \sum_{s'\in S}\sum_{a\in A} T(s|s',a) y_{s',a} = \alpha(s) \quad \forall\, s \in S; \tag{5.2.24}$$

$$\sum_{s\in S}\sum_{a\in A} a_i \cdot y_{s,a} \leq x_i \quad \forall\, i \in V; \tag{5.2.25}$$

$$x_i \in \mathbb{N}_+ \quad \forall\, i \in V; \tag{5.2.26}$$

$$u_{i,j} \in \{0,1\} \quad \forall\, i \in V^D, j \in V^I; \tag{5.2.27}$$

$$y_{s,a} \geq 0 \quad \forall\, s \in S, a \in A. \tag{5.2.28}$$

5.3 Case Study: Knoxville, Tennessee

In this section, we first present the data used in the case study, as well as methods for estimating model parameters. Then, we provide an alternated NLP model specifically formulated for the case study. The model features additional constraints that depict realistic connections between the considered CIS facilities. Finally, we apply the previously develop decomposition method to the formulation as a more efficient solution algorithm.

5.3.1 Data & Parameter Estimation

We collect real-world CIS data from the city of Knoxville, Tennessee, according to the Homeland Infrastructure Foundation-Level Data (HIFLD) (U.S. Department of Homeland Security 2022). Specifically, geographic information system (GIS) coordinates and service capacities of 59 CIS facilities from five categories are acquired from HIFLD, including 22 electric substations, 7 cellular towers, 4 hospitals, 9 police stations, and 17 fire stations.

Figure 5.2 shows the GIS locations of all facilities. Among the CIS facilities, cellular towers, hospitals, police stations, fire stations, and manufacturing companies all require electricity. We consider electricity substations as independent CIS facilities and others as dependent CIS facilities. The cost of constructing overhead electricity transmission lines is reported to be $285,000 per mile (Public Service Commission of Wisconsin 2021). The distance between CIS facilities is calculated using real-word street distances from the Google Maps API (Google 2022). The cost of unit defense resources is estimated using the daily salary ($190) of security guards in Tennessee (CareerExplorer 2022).

Outputs of all types of CIS facilities are estimated using real-world data, or data from the literature. The electric substations are considered to serve the total population, 190,740, in Knoxville as customers (U.S. Census Bureau 2020). Customers are divided and assigned to each substation according to their capacity. Economic values of substations are estimated using the $2.70 cost of loss of electricity service per customer per hour (Salman and Li 2018). In the following, all economic costs are calculated as daily costs. The output of an electric substation is calculated by

$$\text{Output} = \text{Customer served} \times \text{Daily cost of loss of service.} \tag{5.3.1}$$

Outputs from cellular towers are estimated using a similar method. The total population in Knoxville is evenly divided by each tower. The daily cost of loss of service is calculated from the literature to be $5.04 (Conrad et al. 2006). Output of a cellular tower is calculated by Equation (5.3.1).

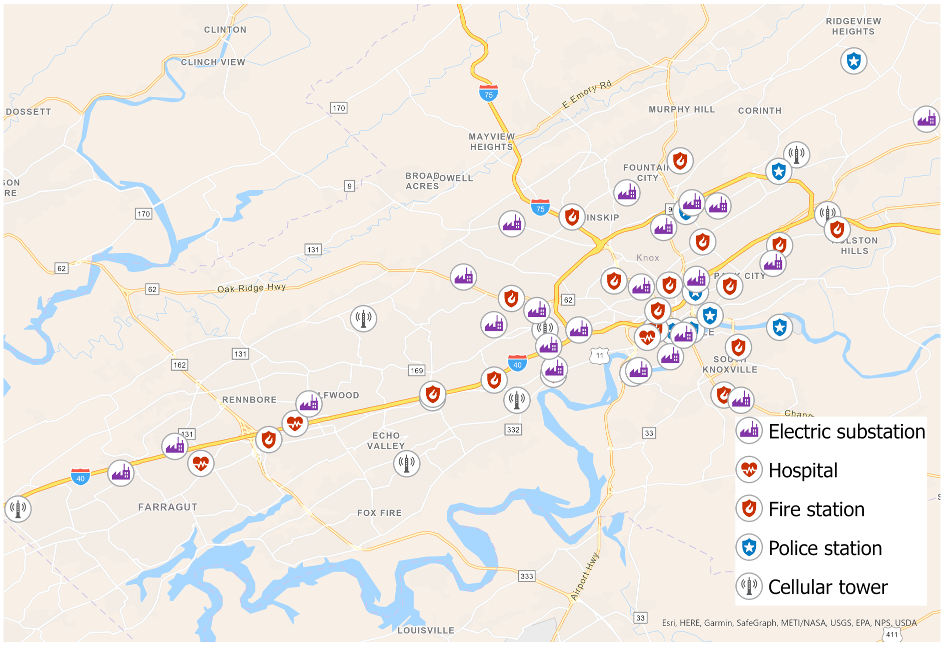


Figure 5.2: GIS locations of 59 CIS facilities in Knoxville, Tennessee.

To estimate the outputs of hospitals, we first identify the hourly increase in mortality rate during power outage to be 0.43 (Apenteng et al. 2018). The value of the quality-adjusted life-year (QALY) is estimated to be $ 265,345 per year (Hirth et al. 2000). The average lifespan in the U.S. is 77 years (Murphy et al. 2021), and the average age of hospital patients is 62 years (Sun et al. 2018). Value of patient life is then calculated as $(77-62)\times 265,345 = \$3,980,175$. We then consider the number of beds in each hospital, as well as the average 0.65 occupation rate of hospital beds in the U.S. (Centers for Disease Control and Prevention 2017). The hourly output is finally converted to daily output. The output of a hospital is calculated by

$$\begin{aligned}\text{Output} = \text{Hourly increase in mortality rate} &\times \text{Value of patient life} \\ &\times \text{Occupation rate} \times \text{Number of beds} \times 24. \end{aligned} \tag{5.3.2}$$

Outputs of the police stations are calculated using crime data from Knoxville (Tennessee Bureau of Investigation 2022). We obtained the yearly crime rates of 13 types of crimes, including murder, rape/sexual assault, assault, robbery, arson, larceny/theft, motor vehicle theft, household burglary, embezzlement, fraud, stolen property, forgery/counterfeiting, and vandalism. Yearly crime rates are then converted into daily crime rates. The cost of each crime is estimated from the literature (McCollister et al. 2010). The population of Knoxville is evenly divided among all police stations. The output of a police station is calculated by

$$\text{Output} = \text{Population served} \times \sum_{\text{All crimes}} \Big(\text{Cost of crime} \times \text{Daily rate of crime}\Big). \tag{5.3.3}$$

Output of fire stations considers the occurrence rate of fire per person, which is estimated to be 0.004149 per year (Haynes and Stein 2017). The number is estimated to increase by 300% during power outage (Federal Emergency Management Agency 2020). Population of Knoxville is evenly divided among all fire stations. The cost of a fire is calculated by adding the property cost per fire, $ 16,610 (Ericson and Lisell 2020), and the casualty cost per fire. The casualty cost per fire is further calculated by multiplying the casualty rate, 0.0256 (U.S. Fire Administration 2022), with the cost of human life, $ 7,500,000 (Federal Emergency

Management Agency 2020). The output of a fire station is calculated using

$$\text{Output} = 300\% \times \text{Daily occurrence rate} \times \text{Served population} \times \text{Cost per fire}. \qquad (5.3.4)$$

Results of the parameter estimation are shown in Table 5.1. The data from five types of CIS facilities are summarized. Averaged output values are provided. The intuitive results suggest that the hospitals are the most valuable CIS facilities, since their failure could cause losses of human life. Electric substations are also essential, because the output of each substation also takes into account the economic cost to residential and commercial users. Fire stations are estimated to have the lowest output due to their superior number and coverage, where the failure of each station only impacts a small neighborhood.

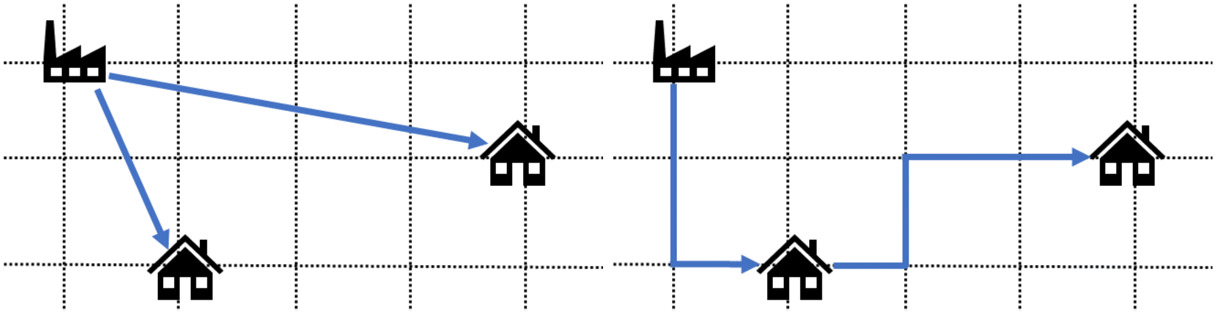
5.3.2 Modified Model Formulation

The case study features the connections between electricity distribution substations and different utilities, which are not fully captured in the original formulation (5.2.18)–(5.2.28). The variables $\boldsymbol{u}$ in the formulation demand that all connections must originate from an independent CIS facility, but electricity can be distributed from any facility, as long as the demand node is connected to the grid. Figure 5.3 demonstrates the differences between the two types of connections.

In order to model the grid connections in more detail, we modify the formulation (5.2.18)–(5.2.28). Note that the original formulation still provides a more generic modeling guideline that can be extended to applications other than this specific case study. We redefine the variable $u_{i,j}, \forall\, i \in V, j \in V^I, j \neq i$ to represent that substation j provides power for facility i, but i and j are not necessarily connected. In the case where i is also a substation, the variable $u_{i,j} = 1$ means that electricity is distributed through i, rather than providing electricity to i. We further define variable $\psi_{i,j}, \forall\, (i,j) \in E$ to represent the physical connection from j to i. Note that we distinguish $\psi_{i,j}$ and $\psi_{j,i}$ to be different variables in order to avoid subtours

Table 5.1: Estimated output from five types of CIS facilities.

CIS facility	Number	Dependency	Average Output ($/day)
Electric substation	22	Independent	341,496
Cellular tower	7	Dependent	138,480
Hospital	4	Dependent	1,386,744
Police station	9	Dependent	146,304
Fire station	17	Dependent	79,800



(a) Direct connections from independent CIS. (b) Power grid connections from other facilities.

Figure 5.3: A comparison between two types of CIS interconnectivity.

in the connections. Then, the following constraints are added to the model:

$$
\begin{aligned}
& u_{i,j} \leq \psi_{j,i} + \sum_{h \in \mathcal{N}(i), h \neq j} u_{h,j} \psi_{h,i} \quad \forall\ i \in V, j \in V^I, j \neq i; \\
& M' \cdot u_{i,j} \geq \psi_{j,i} + \sum_{h \in \mathcal{N}(i), h \neq j} u_{h,j} \psi_{h,i} \quad \forall\ i \in V^D, j \in V^I, j \neq i; \\
& \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}, j \neq i} \psi_{i,j} \leq |\mathcal{S}| - 1 \quad \forall\ \mathcal{S} \subset V,
\end{aligned}
$$

where $\mathcal{N}(i)$ represent the set of neighbors of $i \in V$, M' is a large number and $\mathcal{S}$ are all subsets in V. The first two constraints guarantee that there exists a physical connection for each pair of interconnected CIS facilities. The third constraints eliminate subtours.

Further changes are made in the objective and the constraints to reflect the realistic grid connections. The modified NLP formulation is show as follows.

$$
\begin{aligned}
\text{NLP}_{\text{CIS}} := \max \quad & -\sum_{i \in V} c^r x_i - \sum_{(i,j) \in E} c^s_{i,j} \psi_{i,j} + \sum_{s \in S} \sum_{a \in A} R_{s,a} y_{s,a} && (5.3.5) \\
\text{s.t.} \quad & \sum_{j \in V^I} u_{i,j} \geq 1 \quad \forall\ i \in V^D; && (5.3.6) \\
& u_{i,j} \leq \psi_{j,i} + \sum_{h \in \mathcal{N}(i), h \neq j} u_{h,j} \psi_{h,i} \quad \forall\ i \in V, j \in V^I, j \neq i; && (5.3.7) \\
& M' \cdot u_{i,j} \geq \psi_{j,i} + \sum_{h \in \mathcal{N}(i), h \neq j} u_{h,j} \psi_{h,i} \quad \forall\ i \in V, j \in V^I, j \neq i; && (5.3.8) \\
& \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}, j \neq i} \psi_{i,j} \leq |\mathcal{S}| - 1 \quad \forall\ \mathcal{S} \subset V; && (5.3.9) \\
& R(s,a) = \sum_{i \in V^I} s_i \cdot r^I_i + \sum_{i \in V^D} \Big[1 + (\sigma_{s,i} - 1) \cdot \delta_i \Big] \cdot s_i \cdot r^D_i \quad \forall\ s \in S, a \in A; && \\
& && (5.3.10) \\
& M' \cdot \sigma_{s,i} \geq \sum_{j \in V^I} s_j \cdot u_{i,j} \quad \forall\ s \in S, i \in V^D; && (5.3.11) \\
& \sigma_{s,i} \leq \sum_{j \in V^I} s_j \cdot u_{i,j} \quad \forall\ s \in S, i \in V^D; && (5.3.12)
\end{aligned}
$$

$$\sum_{a\in A} y_{s,a} - \gamma \sum_{s'\in S}\sum_{a\in A} T(s|s',a) y_{s',a} = \alpha(s) \quad \forall\, s \in S; \tag{5.3.13}$$

$$\sum_{s\in S}\sum_{a\in A} a_i \cdot y_{s,a} \le x_i \quad \forall\, i \in V; \tag{5.3.14}$$

$$x_i \in \mathbb{N}_+ \quad \forall\, i \in V; \tag{5.3.15}$$

$$u_{i,j} \in \{0,1\} \quad \forall\, i \in V^D, j \in V^I; \tag{5.3.16}$$

$$\psi_{i,j} \in \{0,1\} \quad \forall\, (i,j) \in E; \tag{5.3.17}$$

$$y_{s,a} \ge 0 \quad \forall\, s \in S, a \in A. \tag{5.3.18}$$

Note that the number M' and M are two different larger numbers where $M' < M$, since constraints with M' only require $M' \ge \max\{|\mathcal{N}(i)| : i \in V\} \cup \{|V^I|\}$.

5.3.3 Applying The Decomposition Method

Although NLP_{CIS} does not contain nonlinear terms in the MDP constraints, it is still a difficult model to solve, considering the integer variables and the large state and action spaces from the MDP. Here, we apply the methods developed in the previous chapters and solve NLP_{CIS} exactly using discretization and decomposition.

Note that in NLP_{CIS}, the MDP variable $\boldsymbol{y}$ is constrained by another variable $\boldsymbol{x}$, which denotes the initial defense resource allocation as one of the strategic decisions. Thus, by considering $\boldsymbol{x}$ as the variable budgets, we apply the TSBD-VB method as an alternate solution algorithm. In this case, only $\boldsymbol{u}$ remains as the first-stage variables, which denotes the service options between dependent and independent CIS facilities. When discretizing the strategic decisions, each resulting MDP model $k = 1, \ldots, K$ represents a possible service scenario with $u_{k,i,j} \in \{0,1\}$, $\forall\, i \in V, j \in V^I, j \ne i$, such that

$$\sum_{j\in V^I} u_{k,i,j} \ge 1 \quad \forall\, i \in V^D, k = 1 \ldots, K. \tag{5.3.19}$$

As such, $u_{k,i,j}$ become parameters to the model rather than variables. Accordingly, we can calculate each $R_{k,s,a}$ using $\boldsymbol{u}$, so that $R_{k,s,a}$ no longer cause nonlinearity in the objective. We let $z_k \in \{0,1\}$, $k = 1, \dots, K$ denote which CIS connectivity decision to choose, and V the objective of the MDP. The integer model can be written in the following form.

$$\mathrm{INT}_{\mathrm{CIS}} := \max \quad -\sum_{i \in V} c^r x_i - \sum_{(i,j) \in E} c^s_{i,j} \psi_{i,j} + V \tag{5.3.20}$$

$$\text{s.t.} \quad u_{k,i,j} \leq \psi_{j,i} + \sum_{h \in \mathcal{N}(i), h \neq j} u_{k,h,j} \psi_{h,i} + M'(1 - z_k) \quad \forall\, i \in V, j \in V^I, j \neq i, k = 1 \dots, K; \tag{5.3.21}$$

$$M'(1 - z_k) + M' \cdot u_{k,i,j} \geq \psi_{j,i} + \sum_{h \in \mathcal{N}(i), h \neq j} u_{k,h,j} \psi_{h,i} \quad \forall\, i \in V, j \in V^I, j \neq i, k = 1 \dots, K; \tag{5.3.22}$$

$$\sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}, j \neq i} \psi_{i,j} \leq |\mathcal{S}| - 1 \quad \forall\, \mathcal{S} \subset V; \tag{5.3.23}$$

$$V \leq \sum_{s \in S} \sum_{a \in A} R_{k,s,a} y_{k,s,a} + M(1 - z_k) \quad \forall\, k = 1 \dots, K; \tag{5.3.24}$$

$$\sum_{a \in A} y_{k,s,a} - \gamma \sum_{s' \in S} \sum_{a \in A} T(s|s', a) y_{k,s',a} = \alpha(s) z_k \quad \forall\, s \in S, k = 1 \dots, K; \tag{5.3.25}$$

$$\sum_{s \in S} \sum_{a \in A} a_i \cdot y_{k,s,a} \leq x_i + M(1 - z_k) \quad \forall\, i \in V, k = 1 \dots, K; \tag{5.3.26}$$

$$x_i \in \mathbb{N}_+ \quad \forall\, i \in V; \tag{5.3.27}$$

$$u_{i,j} \in \{0,1\} \quad \forall\, i \in V^D, j \in V^I; \tag{5.3.28}$$

$$\psi_{i,j} \in \{0,1\} \quad \forall\, (i,j) \in E; \tag{5.3.29}$$

$$y_{s,a} \geq 0 \quad \forall\, s \in S, a \in A. \tag{5.3.30}$$

In the formulation, $y_{k,s,a}$ are given an extra dimension to include all K MDP models.

Next, we decompose $\mathrm{INT}_{\mathrm{CIS}}$ and derive the necessary elements for the TSBD-VB method. In Step-I, according to the generic model in Chapter 3.5.2, we formulate the following MP:

$$\min \quad \sum_{k=1}^{K} \sum_{s \in S} \alpha(s) \theta_{k,s} \tag{5.3.31}$$

$$\text{s.t.} \quad -\rho_{k,i} \geq -c^r \quad \forall\, i \in V, k = 1, \ldots, K; \tag{5.3.32}$$

$$\theta_{k,s} \text{ unrestricted} \quad \forall\, s \in S, k = 1, \ldots, K; \tag{5.3.33}$$

$$\rho_{k,i} \geq 0 \quad \forall\, i \in V, k = 1, \ldots, K. \tag{5.3.34}$$

In the MP, $\boldsymbol{\theta}$ represents the value of states for each MDP, and $\boldsymbol{\rho}$ is the dual variable corresponding to the resource constraints. Given the solution $(\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\rho}})$ to the MP, the dual of SP for state s and model k can be written as

$$\max \quad \sum_{a \in A} \Big[R_{k,s,a} + \gamma \sum_{s' \in S} T(s'|s,a)\bar{\theta}_{k,s'} - \sum_{i \in V} a_i \rho_{\bar{k},i}\Big] \cdot \mu_{k,s,a} \tag{5.3.35}$$

$$\text{s.t.} \quad \sum_{a \in A} \mu_{k,s,a} = 1; \tag{5.3.36}$$

$$\mu_{k,s,a} \geq 0 \quad \forall\, a \in A. \tag{5.3.37}$$

From the dual problem, the optimality cuts and the convergence values can be easily formulated using previously established results. Then, the K-MCLD algorithm applies to obtain the optimal variables $\boldsymbol{\mu}^*$ and $\boldsymbol{\rho}^*$.

Note that since $x_i \in \mathbb{N}_+$ are integer variables, the decomposition in the second stage does not calculate the value of $\boldsymbol{x}$ directly, since models with integer variables do not always satisfy strong duality. Thus, in Step-II, we derive an alternate approach to calculating resource usages. Since $\boldsymbol{x}$ is constrained by the original MDP variable $\boldsymbol{y}$, we first derive the value of the occupation measure $\boldsymbol{y}$ from the decomposition model. Let $Y_{k,s} = \sum_a y_{k,s,a}$, $\forall\, s \in S, k = 1, \ldots, K$. We have

$$\mu^*_{k,s,a} = \frac{y_{k,s,a}}{Y_{k,s}} \quad \Rightarrow \quad y_{k,s,a} = \mu^*_{k,s,a} \cdot Y_{k,s}. \tag{5.3.38}$$

Using the feasibility of $\boldsymbol{y}$, the following system of equations can be derived w.r.t $\boldsymbol{Y}$:

$$Y_{k,s} - \gamma \sum_{s'} \Big(T(s|s',a)\mu^*_{k,s',a}\Big) \cdot Y_{k,s} = \alpha(s) \quad \forall\, s \in S, k = 1, \ldots, K,$$

where $\boldsymbol{\mu}^*$ is the optimal dual variables for the decomposition model. In the above system, $|S| \cdot K$ equations corresponds to $|S| \cdot K$ variables. Trivially, solving the above system provides the values of $\boldsymbol{Y}$, and thus $\boldsymbol{y}$.

Utilizing the optimal variables $\boldsymbol{\mu}^*$ and $\boldsymbol{\rho}^*$ from Step-I, the integer model in Step-II can be formulated as follows.

$$\begin{aligned}
\max \quad & -\sum_{i \in V} c^r x_i - \sum_{(i,j) \in E} c^s_{i,j} \psi_{i,j} + V && (5.3.39)\\
\text{s.t.} \quad & u_{k,i,j} \leq \psi_{j,i} + \sum_{h \in \mathcal{N}(i), h \neq j} u_{k,h,j} \psi_{h,i} \\
& \qquad + M'(1 - z_k) \quad \forall\ i \in V, j \in V^I, j \neq i, k = 1 \ldots, K; && (5.3.40)\\
& M'(1 - z_k) + M' \cdot u_{k,i,j} \geq \psi_{j,i} \\
& \qquad + \sum_{h \in \mathcal{N}(i), h \neq j} u_{k,h,j} \psi_{h,i} \quad \forall\ i \in V, j \in V^I, j \neq i, k = 1 \ldots, K; && (5.3.41)\\
& \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}, j \neq i} \psi_{i,j} \leq |\mathcal{S}| - 1 \quad \forall\ \mathcal{S} \subset V; && (5.3.42)\\
& V \leq \sum_{s \in S} \alpha(s) \theta_{k,s} + M(1 - z_k) \quad \forall\ s \in S, a \in A, k = 1 \ldots, K; && (5.3.43)\\
& \theta_{k,s} \leq \sum_{a \in A} \mu^*_{k,s,a} \cdot \Big[R_{k,s,a} + \gamma \sum_{s'} T(s'|s,a) \theta_{k,s'} \\
& \qquad - \sum_{i \in V} a_i \rho^*_{k,i} \Big] \quad \forall\ s \in S, k = 1 \ldots, K; && (5.3.44)\\
& Y_{k,s} - \gamma \sum_{s'} \Big(T(s|s',a) \mu^*_{k,s',a} \Big) \cdot Y_{k,s} = \alpha(s) \quad \forall\ s \in S, k = 1, \ldots, K; && (5.3.45)\\
& \sum_{s \in S} \sum_{a \in A} a_i \cdot \mu^*_{k,s',a} \cdot Y_{k,s} \leq x_i + M(1 - z_k) \quad \forall\ i \in V, k = 1 \ldots, K; && (5.3.46)\\
& x_i \in \mathbb{N}_+ \quad \forall\ i \in V; && (5.3.47)\\
& u_{i,j} \in \{0,1\} \quad \forall\ i \in V^D, j \in V^I; && (5.3.48)\\
& \psi_{i,j} \in \{0,1\} \quad \forall\ (i,j) \in E; && (5.3.49)\\
& \theta_{k,s}, Y_{k,s} \geq 0 \quad \forall\ s \in S, k = 1, \ldots, K. && (5.3.50)
\end{aligned}$$

5.4 Case Study: Experiments & Results

In this section, we conduct numerical experiments to validate the proposed models and algorithms. According to preliminary results, NLP_{CIS} becomes intractable to solve for large instances. Thus, we first consider a small instance of six CIS facilities (CIS-6), sampled from the collected data. We further conduct sensitivity analysis on CIS-6 to show that the model is applicable to different scenarios. Then, we compare algorithm performances by varying the instance size, not only to show that the TSBD-VB method produces true optimal solutions, but using the decomposition algorithm is also an efficient way of solving the problem.

5.4.1 Baseline Model

The CIS-6 instance consists of six CIS facilities, including two electric substations, a cellular tower, a hospital, a police station, and a fire station. Figure 5.4 shows the geographic configuration of CIS-6. The instance features closely adjacent CIS facilities near the campus of the University of Tennessee. Outputs of CIS facilities are consistent with the estimations from Chapter 5.3.1. In the following, we show the results of CIS-6, calculated from NLP_{CIS}, as a baseline model.

Specifically, we consider two attack/defense intensities, i.e., $a_i, \beta_i \in \{1, 2\}$, $\forall\ i \in V$. We let $\delta_i = 0.8$, $\forall\ i \in V^D$, suggesting that 80% of the output from dependent CIS facilities requires connection to an independent CIS facility. We use $\gamma = 0.7$ to model the situation where the attacks stop after around 10 attacks, since $0.7^{10} = 0.028$ heavily discounts the state values. Attack probability is proportional to CIS facility output, representing that deliberate terrorist attacks are more likely to concentrate on high-value targets. The NLP_{CIS} model is solved using Gurobi, which still takes around 20 seconds to find the optimal solution, with negligible gaps ($< 1 \times 10^{-5}$).

The results of the model are summarized in Table 5.2. The total objective of the model is around \$8.34 million, representing the total economic benefit of the CIS facilities to the society subject to the intentional attacks. The objective is calculated by subtracting the construction cost of electricity distribution lines, \$2.23 million, and the cost of defense

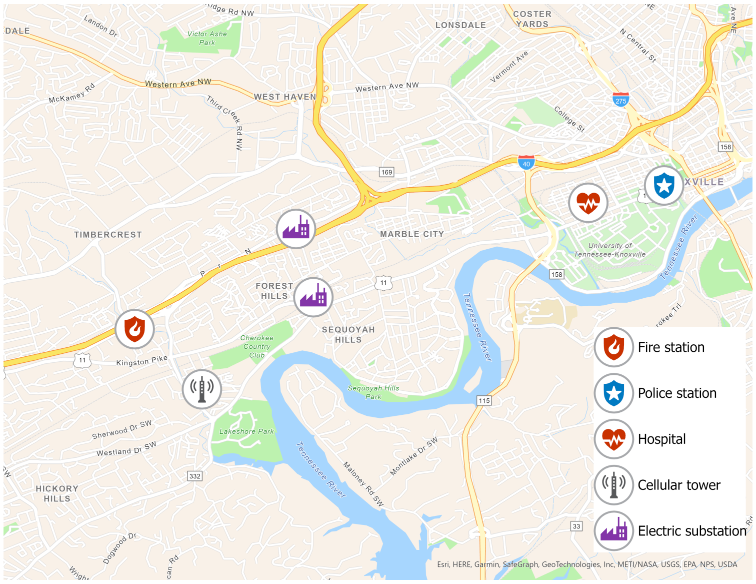


Figure 5.4: A small instance of six CIS facilities in Knoxville, Tennessee.

Table 5.2: Monetary values for CIS-6.

Total objective ($)	Electricity line cost ($)	Resource cost ($)	Total output ($)
8,338,656	-2,225,499	-7,031	10,571,187

resources, \$7,031, from the total facility output during the attack period, \$10.57 million. Overall, although strategic planning introduces high costs for the decision maker, it also guarantees approximately 85% of CIS output remains undisturbed during prolonged attacks.

To clearly convey the results, we illustrate the constructed electricity distribution lines and CIS facility services (dependencies) in Figure 5.5. The model connects all facilities through a parsimonious solution. Instead of connecting all dependent facilities to independent facilities, the model uses the cellular tower, the hospital, and substation 2 as media to connect the fire station, the police station, and substation 1 into the grid. Since all facilities are connected, both substations 1 and 2 serve all four utilities, so that when one of them is disabled, the utilities still generate outputs through the other.

In addition, Figure 5.6 shows the number of allocated resources at each facility, and the corresponding facility output. The figure demonstrates that our model makes rational and intuitive decisions. The most defense resources are allocated to the hospital and substation 2, since they generate the most output. A relatively large amount of resources is also allocated to substation 1, even though its output is not significantly higher than the others, because the model understands that once substation 1 is disabled, all the dependent facilities could face dysfunction.

To illustrate the policy of the MDP, we plot the average defense intensity for each CIS facility in Figure 5.7. The figure represents the average amount of defense resources used to protect a facility against an attack. Similar to previous results, the model emphasizes on protecting substation 2 and the hospital, by using more resources compared with the other facilities. Although the hospital generates more output than substation 1, the model still uses more resources to defend substation 1, since it is connected to all dependent CIS facilities.

5.4.2 Sensitivity Analysis

In this section, we conduct sensitivity analysis by evaluating the results under different parameters. The analysis not only shows the degree of model sensitivity to parameters, but

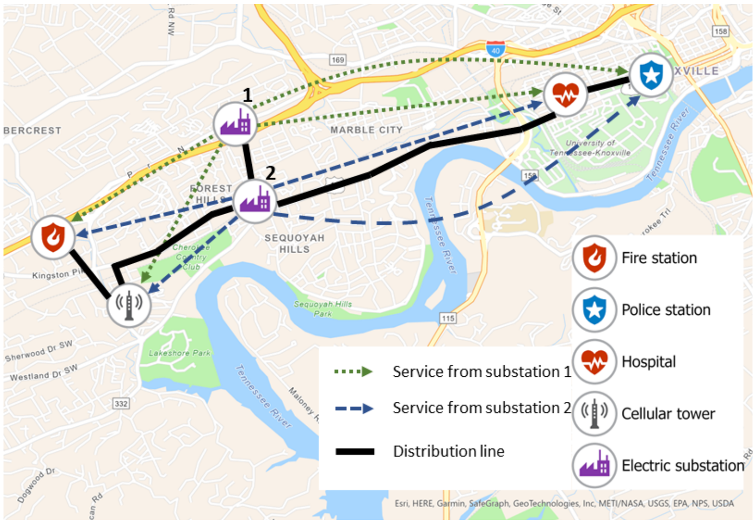


Figure 5.5: Electricity line constructions and CIS dependencies for CIS-6.

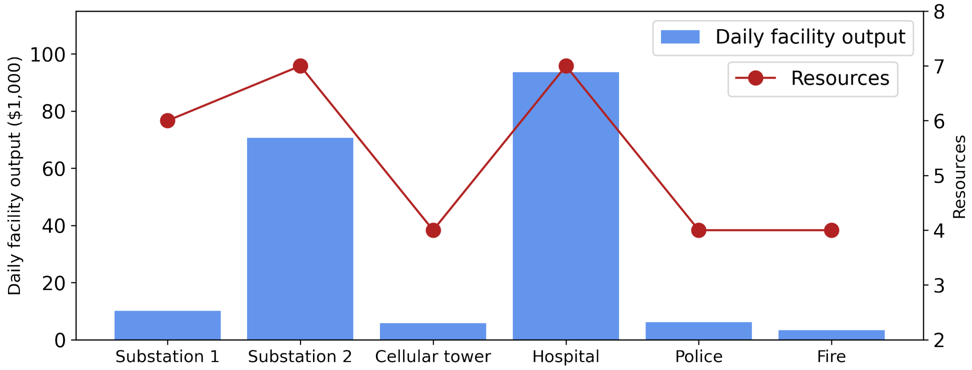


Figure 5.6: Resource allocation decisions for CIS-6.

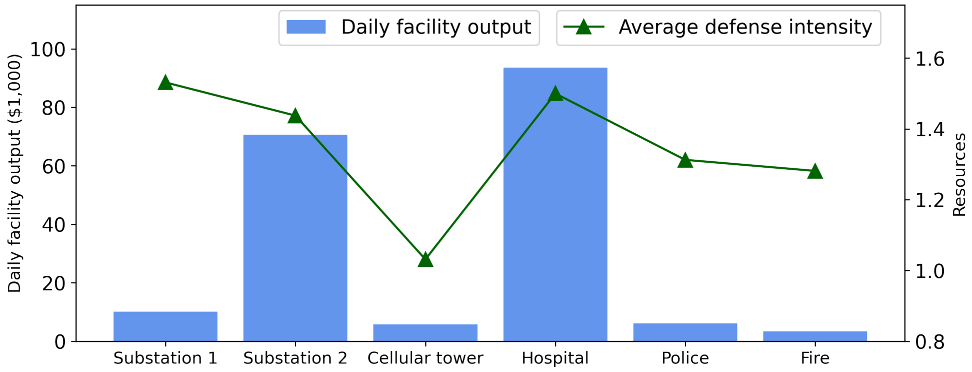


Figure 5.7: Average defense intensity from the optimal policy.

also represents the defender's perception of the attacks, where different model configurations represent different attack scenarios. By analyzing the model's behavior, we provide managerial insights to the decision maker regarding the optimal strategic and operational defense measures.

Discount factor First, we modify the discount factor γ. In the baseline model, $\gamma = 0.7$ discounts facility rewards heavily after around 10 attacks, suggesting the decision maker's belief about the total number of attacks. We extend the current results by considering four more alternatives, with $\gamma = 0.4$, 0.7, 0.9, 0.99 and 0.999, where $\gamma = 0.4$ features short attack periods, and $\gamma = 0.999$ features prolonged attacks with nearly "infinite" attacks.

Table 5.3 shows the results of CIS-6 under different discount factors. As expected, longer planning periods correspond to larger economic values, since the CIS facilities continue to generate outputs. The electricity line construction costs remain the same across all discount factors, suggesting the model provides robust strategic decisions about CIS network design, no matter the planning horizon. Intuitively, defense resource costs increase as the discount factor grows larger, because more resources are required to counter the attacks on longer horizon. We have also calculated the output in ideal situations, i.e., what the facility output should have been if there were no attacks. As the table suggests, with a longer period of planning, heavier losses are imposed on the CIS facilities.

Contest function In Table 5.3, $\gamma = 0.99$ and 0.999 show similar results, suggesting that the model demonstrates a "converged" state, where increasing the planning period does not significantly increase the total output. This can be explained by the contest function (5.2.7), using which the defender at best has 67% chance to win the contest under the current model setting with $a_i, \beta_i \in \{1, 2\}$. To consider the scenario that defense resources are more efficient in protecting high-value CIS facilities, we modify the contest function and include an efficiency multiplier λ for the defender:

$$P(\text{attack success}) = \frac{\beta}{\beta + \lambda a}. \tag{5.4.1}$$

Table 5.3: Monetary values for CIS-6 under different discount factors.

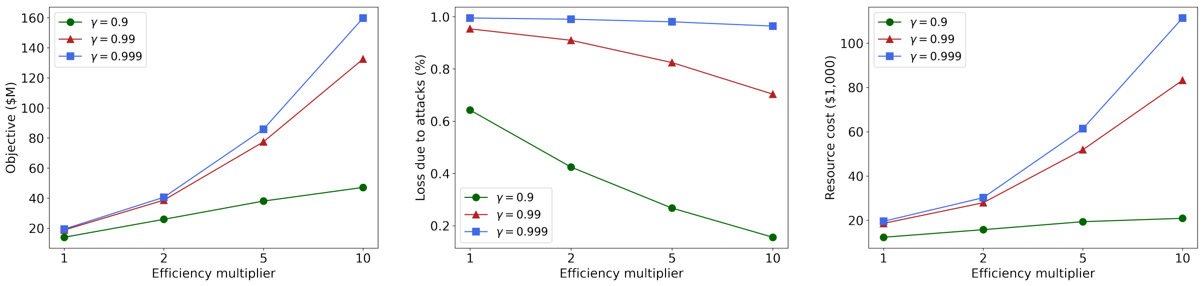
γ	Total objective (\$M)	Electricity line cost (\$M)	Resource cost (\$)	Total output (\$M)	Ideal output* (\$M)	Loss due to attacks† (%)
0.4	4.67	-2.23	-3,421	6.55	7.58	13.59
0.7	8.34	-2.23	-7,031	10.57	15.17	30.32
0.9	14.02	-2.23	-12,352	16.26	45.49	64.26
0.99	18.82	-2.23	-18,624	21.06	454.96	95.37
0.999	19.44	-2.23	-19,574	21.69	4549.68	99.52

*: CIS facility output without any attacks. †: calculated as (ideal output - total output) / ideal output.

We further conduct experiments by selecting λ from the list $[1, 2, 5, 10]$, representing that the defender considers the resources with increased efficiency in protecting CIS facilities. We choose λ to be as large as 10 so that the best defense probability is as high as around 80%. To compare with previous results, we consider discount factors $\gamma = 0.9$, 0.99 and 0.999.

Figure 5.8 shows the results of the model for $\lambda = 1$, 2, 5, and 10. Results are shown from three fronts, the total objective values, the losses in facility output due to attacks, and the costs for defense resources. The total objective values of three models with $\gamma = 0.9, 0.99$, and 0.999 are shown in Figure 5.8a. With increased defense resource efficiency, objective values increase, since the long-term output from CIS facilities can be well protected. As a result, the facility output losses due to attacks reduce with the efficiency. Figure 5.8b shows the percentage losses compared with the situations without attacks. Note that for $\gamma = 0.999$, the reduction is not significant, because the planning horizon is too long to prevent facility failure from intentional attacks. Finally, Figure 5.8c shows resource costs in the strategic planning phase. The increased resource costs partly contribute to the increased total objective values and the decreased losses, allowing the MDP policy to provide better defense strategies for the CIS facilities.

Output dependency Next, we change the coefficient δ_i, $\forall\ i \in V^D$, representing the proportion of output for a CIS facility that depends on other facilities. We let $\delta_i = 0.2, 0.4, 0.6, 0.8,$ and 1.0, $\forall\ i \in V^D$. Other parameters are consistent with the baseline model. Figure 5.9 shows the facility output and resource costs under different δ. The facility output shows interesting trends. When dependency reduces from high to medium, facility output increases, since it is impacted less by the attacks. However, when the dependency reduces from medium to low, the output also reduces. This is because the dependent facilities are protected with fewer resources, and assigned with lower resource usage in the defense strategy. Thus, when attacks happen, dependent facilities are more likely to lose their entire output under low dependency. Figure 5.9 also shows the resource costs under different δ. All strategies remain the same except for when $\delta = 1.0$, in which case there is no need to defend



(a) Objective value

(b) Output loss

(c) Resource cost

Figure 5.8: Results for CIS-6 under different efficiency multipliers.

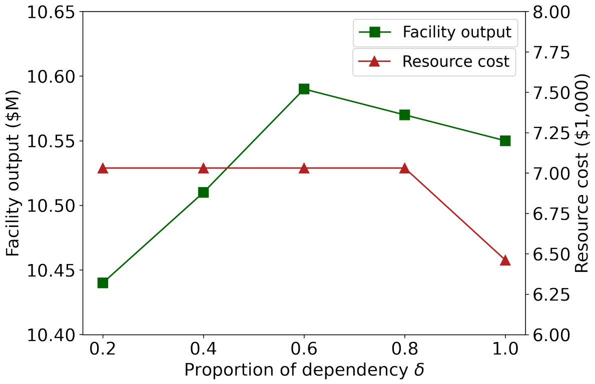


Figure 5.9: Facility output and resource costs for CIS-6 under different δ.

a dependent CIS facility, if the independent facilities fail, since all of its output depends on other facilities.

We further plot the average resource usage in the optimal policy for $\delta = 0.8$ and 1.0, in Figure 5.10. The results are intuitive and confirm our previous argument, where when $\delta = 1.0$, less attention is paid to dependent facilities, but more is paid to independent facilities. Note that although independent facilities are more secured under $\delta = 1.0$, the model purchases less defense resourced in total compared with $\delta = 0.8$, and produced less output facing attacks, as Figure 5.9 suggested.

Thus, when designing CIS facilities, decision makers should not only consider network design or resource allocation, but also ways to reduce the dependency of each CIS facility, such as preparing backup generators, or storing additional emergency resources. As such, even when independent CIS facilities are disabled, other connected utilities still function on their own, increasing the overall robustness of the CIS network.

Resource cost In real applications, defense resources are not limited to security guard salary. Sometimes, the unit cost of defense resources can be expensive, such as state-of-the-art surveillance and sensory systems, or specialized machinery and equipment. Here, we vary the cost of defense resources c^r. From our parameter estimation, $c^r = 190$. We further extend the estimation by considering $c^r = 190, 2,000, 5,000$, and 10,000. Other parameters are consistent with the baseline model.

Figure 5.11 shows the results under different resource costs. The figure suggests that the model is not sensitive to the resource cost. Even though the total objective reduces when the resource becomes more expensive, the MDP policy still manages to maintain the facility output during the attacks. The maximum reduction in facility output is from \$10.57 to \$ 10.51, i.e., around 0.57% decrease, under 50-fold changes in the resource price. Thus, our model provides stable solutions with respect to the defense resource cost.

Attack probability Finally, we extend our results to include different attack patterns. We show that our model not only makes decisions for intentional attacks, but also for natural

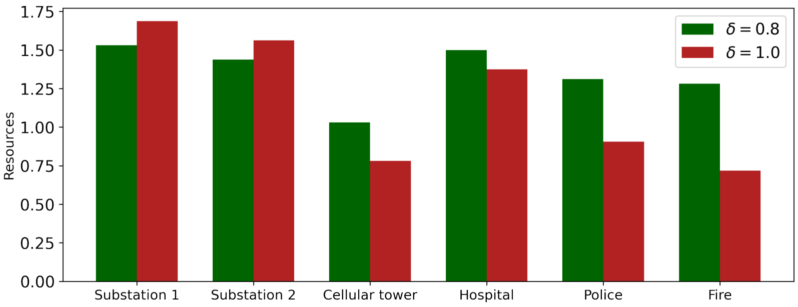


Figure 5.10: Resource usage in the optimal policy for CIS-6 under different δ.

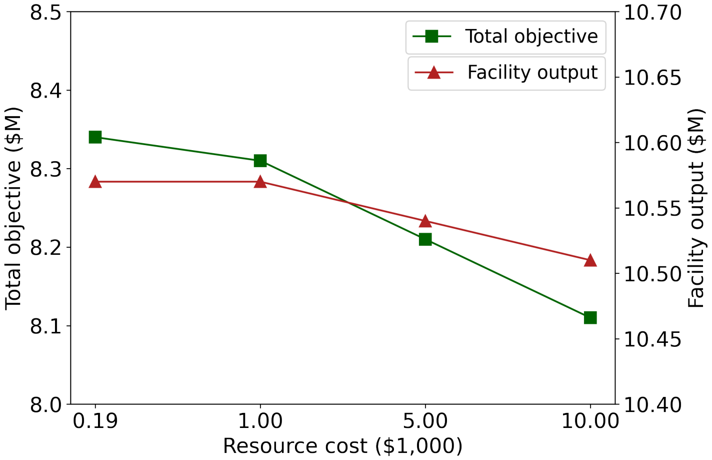


Figure 5.11: Total objective and facility output for CIS-6 under different resource costs.

disasters, in which case the “attack” probability for each CIS facility is unknown to the defender. We use a uniform distribution to model the random attacks. Other parameters are consistent with the baseline model.

Results are shown in Figure 5.12. Results for the random attacks are compared with those from intentional attacks. The figure shows the resource distribution among all facilities. Compared with intentional attacks, for the random attacks, the model places resources more evenly across all facilities, but still emphasizes more on high-valued targets, such as the substations and the hospital.

5.4.3 Algorithm Comparison

The previous experiments are conducted using an instance with only 6 CIS facilities, not only for illustrative purposes, but also because larger instances are too complex to solve in a reasonable time. The complexity of NLP_{CIS} mainly comes from two fronts, the nonlinearity of constraints, and the curse of dimensionality of MDP. For example, in CIS-6, there are $2^6 = 64$ states, and $3^6 = 729$ actions, resulting in 2,985,984 transition probabilities. The CIS-6 instance takes as long as one minute to solve. In the following, we evaluate the performance of the algorithms proposed in Chapter 5.3.3, where the decomposition technique offers a more efficient way of finding optimal solutions.

Specifically, we compare the performance between NLP_{CIS}, INT_{CIS}, and TSBD-VB. The algorithms are tested on three instances, where 5, 6, 7, and 8 CIS facilities are sampled from the datasets. Note that although the number of facilities only increases by one, the resulting state and action dimensions still differ dramatically. Table 5.4 summarizes the configurations for the four instances. To reduce the scales of the instances, we only allow CIS interdependency within a radius. The radius is set to be half of the longest distance between facilities in an instance. Instance parameters mostly follow the CIS-6 model, with the exception that $\gamma = 0.999$, representing the solution to a long-term stationary policy. All experiments are conducted on a Linux server with 2.30GHz Intel Xeon Gold CPU and 256

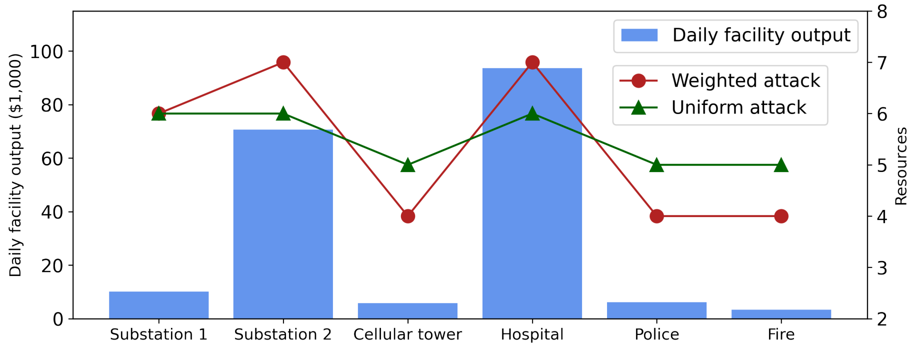


Figure 5.12: Resource allocation for CIS-6 under intentional and random attacks.

Table 5.4: Summary of four CIS instances.

CIS	$\|S\|$	$\|A\|$	$\|S\| \times \|A\|$
5	32	243	7,776
6	64	729	46,656
7	128	2,187	279,936
8	256	6,561	1,679,616

GB memory. The LP models are solved with Gurobi via the Python interface. Time limits of 3600 seconds are imposed when solving LP models.

Results of the algorithm comparison are shown in Table 5.5. Consistent with previous results in Chapter 4.2, in general, the TSBD-VB method shows advantages in finding optimal solutions more efficiently than solving the nonlinear and integer models. For the larger instance with 7 CIS facilities, the TSBD-VB method solves the problem 67.87% faster than NLP_{CIS}. In the case of 8 CIS facilities where both NLP_{CIS} and INT_{CIS} cannot find feasible solutions within one hour, the TSBD-VB method is still able to solve the problem to the true optimum. The TSBD-VB method underperforms on smaller instances compared with NLP_{CIS}, where the cost of additional operations outweighs the benefit of decomposition. Note that different from previous results, the integer model, INT_{CIS}, performs the worst, due to the large number of variables and constraints. However, the integer model still serves as a basis for the decomposition method, which outperforms conventional methods with a "divide-and-conquer" strategy.

5.5 Discussion

In this chapter, we consider a real-world application for the LSSD framework, protecting interconnected CIS facilities from sequential and stochastic attacks. Different from the literature, we model the problem from the defender's perspective, with only partial information about the attacker's intentions. In addition, the defender protects the facilities from sequential attacks with an unknown number of attacks. By modeling the problem using the LSSD framework, we make strategic decisions about the connectivity of the CIS network, and the allocation of defense resources. The infinite-horizon MDP in the second stage allows us to calculate a stationary policy optimal policy, according to which the defender devises defense strategies independent of the number of attacks.

To model the CIS defense problem, we collect real-world data in a middle-size city in the U.S., and conduct thorough parameter estimations. Using previously established theoretical results, we propose a nonlinear model, NLP_{CIS} and an integer model, INT_{CIS}. We further

Table 5.5: Algorithm comparison on four CIS instances.

CIS	NLP$_{\text{CIS}}$			INT$_{\text{CIS}}$			TSBD-VB			Impr.† (%)
	Objective ($M)	Gap (%)	Runtime (s)	Objective ($M)	Gap (%)	Runtime (s)	Objective ($M)	Gap (%)	Runtime (s)	
5	18.21	0.00	0.14	18.21	0.00	3.42	18.21	0.00	0.68	< -100.00
6	30.28	0.00	22.70	30.28	30.28	2870.36	30.28	0.00	12.30	45.81
7	35.20	0.00	675.58	–	100.00	3600.00	35.20	0.00	217.06	67.87
8	–	100.00	3600.00	–	100.00	3600.00	39.14	0.00	3110.18	–

†: improvement is calculated by comparing TSBD-VB with the best between NLP$_{\text{CIS}}$ and INT$_{\text{CIS}}$.

apply the decomposition method TSBD-VB to solve the integer formulation. Algorithms are compared using CIS instances of different sizes. Results show that the TSBD-VB method consistently produces the best performance. The improvements are less significant compared with previous results, due to the differences in problem structures, where states and actions in the CIS problem outnumber those in the testing instances in Chapter 4.2.

The proposed approach for the CIS defense problem is also validated using a smaller instance with 6 CIS facilities. We first show the results of a baseline model, including the CIS interconnectivity and defense resource allocation. The model produces intuitive results, including a parsimonious plan for an electricity distribution grid that connects all dependent facilities with independent ones, and a resource allocation strategy proportional to the magnitude of CIS facility outputs. The generated MDP policy also intelligently focuses more on high-value facilities, such as the hospital, or electricity substations on which other facilities depend.

We have also conducted a sensitivity analysis on five model parameters on top of the baseline model, including the discount factor, contest function, output dependency, resource cost, and attack probability. By varying the discount factor, we show the model's behavior on different planning horizons, from short terms with around 5–10 attacks, to long terms where the model shows a "converged" state, demonstrating the LSSD framework's capability of long-term decision making. The contest function is modified to include an additional parameter that describes the efficiency of the defense resources. Results suggest that with higher efficiency, CIS facilities can be better protected, with increased facility output during the attacks. Another aspect of CIS network design is analyzed by changing the output dependency for all dependent CIS facilities. The results demonstrate the importance of reducing dependency on other facilities, leading to increased robustness of the CIS network. In addition, we have also analyzed the model's sensitivity to the resource cost, since defense resources are not always cheap in practical situations. Results show that the model is not sensitive to the resource cost, i.e., facility outputs are well-protected even when the resource price is increased by 50 folds. Finally, we demonstrate that the model is extensible to many real-world applications by considering choosing targets intentionally, such as terrorist attacks, or randomly, such as natural disasters. Under both scenarios, the model generates meaningful and intuitive strategic plans for allocating resources to protect CIS facilities.

All in all, through this study, we have demonstrated the LSSD framework's capability of solving real-world problems, where strategic and operational decisions for the CIS defense problem are optimized at the same time. In addition, we have shown that the proposed solution methods solve the framework more efficiently compared with conventional methods. Moreover, we have modeled and solved a CIS defense problem that is deemed difficult in the literature, and is of critical importance to social welfare and national security. Further analyses on model parameters have provided insights to practitioners for strategically planning CIS networks and operational management in response to potential failure.

Chapter 6

Conclusion

This dissertation proposes the LSSD framework, which fills the research gap by formulating a generic two-stage mathematical model that jointly optimizes strategic decisions and stochastic operational decisions in the long term. The framework models the strategic decision using LP and the operational decisions using MDP. The decisions of the framework are combined together through the dual of the LP formulation of MDP, resulting in the NLP formulation, an optimization model with nonlinear objectives and constraints. We further analyze the nonlinear model with generalized Benders decomposition, based on which we discretize the first-stage strategic decisions and propose the alternate INT formulation, an optimization model with integer variables.

The LSSD framework is also extended to CMDP and CMDP with variable budgets, to model situations where resources required for taking actions are limited, or multiple objectives need to be satisfied. The extensions are first formulated as nonlinear models, namely NLP-C and NLP-VB, and then discretized into integer models INT-C and INT-VB.

The computational complexity of the LSSD framework and the extensions are briefly discussed, where the integer variables and the curse of dimensionality from MDP prevent the nonlinear and the integer models to be solved in efficient ways. This motivates us to develop novel algorithms that reduce the computational difficulties for the model. First, we apply Benders decomposition to MDP. The LP formulation of MDP is decomposed as an

MP and several SP, each corresponding to a state. The decomposition model is solved using the MCLD algorithm. We further prove mathematical properties of the algorithm to show that it finds the true optimal policy for MDP. The decomposed model is extended to three types of special MDP problems, including MDP with action-free transition probabilities, MDP with the monotone optimal policy, and CMDP.

Applying the developed decomposition algorithm to the LSSD framework, we further propose the TSBD method to solve the integer formulation of the framework to optimality. In Step-I of the TSBD methods, we use the K-MCLD algorithm to find the optimal multipliers for second-stage MDP problems. In Step-II, we construct another integer model and solve it to obtain the optimal strategic decision in the first stage. The TSBD algorithm is extended to CMDP and CMDP with variable budgets.

Computational experiments are conducted to evaluate the performance of the MCLD algorithm and the TSBD method. We adopt four MDP benchmarking problems from the literature and extend them for the LSSD framework. Experiment results show that the MCLD algorithm solved MDP problems up to over 90% faster that the LP formulation and its dual. The MCLD algorithm also outperforms the state-of-the-art exact solution algorithms such as MPI, in long-term decision making with large discount factors. Further analyses suggest that the MCLD algorithm behaves in a collective way of both VI and PI, where the primal problem iterates to find better state values, and the dual problem iterates to find better policies. Further experiments on the LSSD framework show similar improvements by using the decomposition approaches. The TSBD method is compared with the NLP and INT models, with up to over 80% improvements in the computation time.

Finally, we utilize the LSSD framework to solve a real-world CIS protection problem under stochastic and sequential attacks. The problem features network design and resource allocation as the strategic decisions, and different levels of defense intensities as the operational decision to counter the attacks. We model the problem from the defender's perspective, who does not have full knowledge of the attacker's intentions. We have also considered the interconnectivity between different CIS facilities, resulting in complex nonlinear constraints with integer variables in the model. Previously established algorithms are applied to the model. The nonlinear formulation is first discretized into an integer formulation. We then apply decomposition algorithms to the integer model as an alternative solution algorithm.

To validate the model, we collect real-world data from a middle-sized city in the U.S., and estimate model parameters either from real data or from the literature. Algorithms are compared using four different model configurations. The proposed algorithms using the TSBD method outperform conventional nonlinear and integer models by approximately 68%. In addition, sensitivity analysis is conducted on five of the model variables. Model behaviors under different parameters are thoroughly investigated. Discussions and insights are provided for practitioners.

Bibliography

Kizito, Rodney, Zeyu Liu, Xueping Li, and Kai Sun (2021). “Multi-stage Stochastic Optimization of Islanded Utility-Microgrids After Natural Disasters”, pp. 1–41. DOI: 10.13140/RG.2.2.27872.61445.

Torres-Rincón, Samuel, Mauricio Sánchez-Silva, and Emilio Bastidas-Arteaga (2021). “A multistage stochastic program for the design and management of flexible infrastructure networks”. *Reliability Engineering & System Safety* 210, p. 107549.

Li, Qi and Guiping Hu (2020). “Multistage stochastic programming modeling for farmland irrigation management under uncertainty”. *Plos one* 15.6, e0233723.

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

Bertsimas, Dimitris and John N Tsitsiklis (1997). *Introduction to linear optimization*. Vol. 6. Athena Scientific Belmont, MA.

Kazemi Zanjani, Masoumeh, Mustapha Nourelfath, and Daoud Ait-Kadi (2010). “A multi-stage stochastic programming approach for production planning with uncertainty in the quality of raw materials and demand”. *International Journal of Production Research* 48.16, pp. 4701–4723.

Mulvey, John M and Bala Shetty (2004). “Financial planning via multi-stage stochastic optimization”. *Computers & Operations Research* 31.1, pp. 1–20.

Delgado, Felipe, Ricardo Trincado, and Bernardo K Pagnoncelli (2019). “A multistage stochastic programming model for the network air cargo allocation under capacity uncertainty”. *Transportation Research Part E: Logistics and Transportation Review* 131, pp. 292–307.

Ayer, Turgay, Oguzhan Alagoz, and Natasha K Stout (2012). “OR Forum—A POMDP approach to personalize mammography screening decisions”. *Operations Research* 60.5, pp. 1019–1034.

Ruszczyński, Andrzej (2010). “Risk-averse dynamic programming for Markov decision processes”. *Mathematical programming* 125.2, pp. 235–261.

Fan, Jingnan and Andrzej Ruszczyński (2018). “Process-based risk measures and risk-averse control of discrete-time systems”. *Mathematical Programming*, pp. 1–28.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. (2015). "Human-level control through deep reinforcement learning". *nature* 518.7540, pp. 529–533.

Silver, David, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. (2017). "Mastering the game of go without human knowledge". *nature* 550.7676, pp. 354–359.

Birge, John R and François Louveaux (2011). *Introduction to stochastic programming.* Springer Science & Business Media.

Kıbış, Eyyüb Y, İ Esra Büyüktahtakın, Robert G Haight, Najmaddin Akhundov, Kathleen Knight, and Charles E Flower (2020). "A Multistage Stochastic Programming Approach to the Optimal Surveillance and Control of the Emerald Ash Borer in Cities". *INFORMS Journal on Computing.*

Kuhn, Daniel (2008). "Aggregation and discretization in multistage stochastic programming". *Mathematical Programming* 113.1, pp. 61–94.

Yin, Xuecheng and İ Esra Büyüktahtakın (2021). "A multi-stage stochastic programming approach to epidemic resource allocation with equity considerations". *Health Care Management Science*, pp. 1–26.

Howard, Ronald A. (1960). *Dynamic programming and Markov processes.* Cambridge: Technology Press of Massachusetts Institute of Technology.

Bellman, Richard (1957). "A Markovian Decision Process". *Journal of mathematics and mechanics* 6.5, pp. 679–684.

Archibald, Thomas W, CS Buchanan, KIM McKinnon, and Lyn C Thomas (1999). "Nested Benders decomposition and dynamic programming for reservoir optimisation". *Journal of the Operational Research Society* 50.5, pp. 468–479.

Jones, Dean A, Chad E Davis, Mark A Turnquist, and Linda K Nozick (2006). "Physical security and vulnerability modeling for infrastructure facilities". *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06).* Vol. 4. IEEE, pp. 79c–79c.

Cooper, William L and Tito Homem-de Mello (2007). “Some decomposition methods for revenue management”. *Transportation Science* 41.3, pp. 332–353.

Keerthisinghe, Chanaka, Gregor Verbič, and Archie C Chapman (2014). “Evaluation of a multi-stage stochastic optimisation framework for energy management of residential PV-storage systems”. *2014 australasian universities power engineering conference (AUPEC)*. IEEE, pp. 1–6.

Manne, Alan S (1960). “Linear Programming and Sequential Decisions”. *Management science*. Management Science 6.3, pp. 259–267.

d’Epenoux, F (1960). “Sur un probleme de production et de stockage dans l’aléatoire”. *Revue Française de Recherche Opérationelle* 14, pp. 3–16.

Oliver, RM (1960). “A linear programming formulation of some Markov decision processes”. *a meeting of the Institute of Management Sciences and Operations Research Society of America, Monterey, California*.

Derman, Cyrus and Morton Klein (1965). “Some remarks on finite horizon Markovian decision models”. *Operations research* 13.2, pp. 272–278.

de Farias, D. P and B Van Roy (2003). “The Linear Programming Approach to Approximate Dynamic Programming”. *Operations research* 51.6, pp. 850–865.

Altman, Eitan (1999). *Constrained Markov decision processes*. Vol. 7. CRC Press.

Achiam, Joshua, David Held, Aviv Tamar, and Pieter Abbeel (2017). “Constrained policy optimization”. *International conference on machine learning*. PMLR, pp. 22–31.

Vieillard, Nino, Olivier Pietquin, and Matthieu Geist (2019). “On connections between constrained optimization and reinforcement learning”. *arXiv preprint arXiv:1910.08476*.

Benders, J. F (1962). “Partitioning procedures for solving mixed-variables programming problems”. *Numerische Mathematik* 4.1, pp. 238–252.

Dolgov, Dmitri A and Edmund H Durfee (2005). “Stationary deterministic policies for constrained MDPs with multiple rewards, costs, and discount factors”. *IJCAI*. Citeseer, pp. 1326–1331.

Satia, Jay K and Roy E Lave Jr (1973). “Markovian decision processes with uncertain transition probabilities”. *Operations Research* 21.3, pp. 728–740.

Mannor, Shie, Duncan Simester, Peng Sun, and John N Tsitsiklis (2007). “Bias and variance approximation in value function estimates”. *Management Science* 53.2, pp. 308–322.

Bäuerle, Nicole and Ulrich Rieder (2019). “Markov decision processes under ambiguity”. *arXiv preprint arXiv:1907.02347*.

Steimle, Lauren N, David L Kaufman, and Brian T Denton (2021b). “Multi-model Markov decision processes”. *IISE Transactions*, pp. 1–39.

Armony, Mor and Amy R Ward (2010). “Fair dynamic routing in large-scale heterogeneous-server systems”. *Operations Research* 58.3, pp. 624–637.

Boussard, Matthieu and Jun Miura (2011). “Objects search: a constrained mdp approach”. *Workshop Active Percept. Object Search Real World, San Francisco, CA, USA*.

Bhandari, Atul, Alan Scheller-Wolf, and Mor Harchol-Balter (2008). “An exact and efficient algorithm for the constrained dynamic operator staffing problem for call centers”. *Management Science* 54.2, pp. 339–353.

Chen, Qiushi, Turgay Ayer, and Jagpreet Chhatwal (2018). “Optimal m-switch surveillance policies for liver cancer in a hepatitis c–infected population”. *Operations Research* 66.3, pp. 673–696.

Heyman, Daniel P and Matthew J Sobel (2004). *Stochastic models in operations research: stochastic optimization*. Vol. 2. Courier Corporation.

Blumentritt, Tim (2006). “Integrating strategic management and budgeting”. *Journal of business strategy*.

Samuelson, William (1986). “Bidding for contracts”. *Management Science* 32.12, pp. 1533–1550.

Geoffrion, Arthur M (1972). “Generalized benders decomposition”. *Journal of optimization theory and applications* 10.4, pp. 237–260.

Conforti, Michele, Gérard Cornuéjols, Giacomo Zambelli, et al. (2014). *Integer programming*. Vol. 271. Springer.

Buchholz, Peter and Dimitri Scheftelowitsch (2019). “Computation of weighted sums of rewards for concurrent MDPs”. *Mathematical Methods of Operations Research* 89.1, pp. 1–42.

Daoui, Cherki, Mohamed Abbad, and Mohamed Tkiouat (2010). “Exact decomposition approaches for Markov decision processes: A survey”. *Advances in Operations Research* 2010.

Ross, Keith W and Ravi Varadarajan (1991). “Multichain Markov decision processes with a sample path constraint: A decomposition approach”. *Mathematics of Operations Research* 16.1, pp. 195–207.

Abbad, Mohammed and Hatim Boustique (2003). “A decomposition algorithm for limiting average Markov decision problems”. *Operations Research Letters* 31.6, pp. 473–476.

Larach, Abdelhadi, S Chafik, and C Daoui (2017). “Accelerated decomposition techniques for large discounted Markov decision processes”. *Journal of Industrial Engineering International* 13.4, pp. 417–426.

Bai, Aijun, Feng Wu, and Xiaoping Chen (2015). “Online planning for large markov decision processes with hierarchical decomposition”. *ACM Transactions on Intelligent Systems and Technology (TIST)* 6.4, pp. 1–28.

Fu, Jie, Shuo Han, and Ufuk Topcu (2015). “Optimal control in Markov decision processes via distributed optimization”. *2015 54th IEEE Conference on Decision and Control (CDC).* IEEE, pp. 7462–7469.

Chen, Peng and Lu Lu (2013). “Markov decision process parallel value iteration algorithm on GPU”. *2013 International Conference on Information Science and Computer Applications (ISCA 2013).* Atlantis Press, pp. 299–304.

Chafik, Sanaa and Cherki Daoui (2015). “A Modified Value Iteration Algorithm for Discounted Markov Decision Processes”. *Journal of Electronic Commerce in Organizations (JECO)* 13.3, pp. 47–57.

Bertsimas, Dimitris and Velibor V Mišić (2016). “Decomposable markov decision processes: A fluid optimization approach”. *Operations Research* 64.6, pp. 1537–1555.

Kushner, H and Ching-Hui Chen (1974). “Decomposition of systems governed by Markov chains”. *IEEE transactions on Automatic Control* 19.5, pp. 501–507.

Dean, Thomas and Shieu-Hong Lin (1995). “Decomposition techniques for planning in stochastic domains”. *IJCAI.* Vol. 2. Citeseer, p. 3.

Dantzig, George B and Philip Wolfe (1960). “Decomposition principle for linear programs”. *Operations research* 8.1, pp. 101–111.

Rebennack, Steffen (2016). “Combining sampling-based and scenario-based nested Benders decomposition methods: application to stochastic dual dynamic programming”. *Mathematical Programming* 156.1-2, pp. 343–389.

Dimitrov, Nedialko B and David P Morton (2009). “Combinatorial design of a stochastic Markov decision process”. *Operations Research and Cyber-Infrastructure*. Springer, pp. 167–193.

Regan, Kevin and Craig Boutilier (2012). “Regret-based reward elicitation for Markov decision processes”. *arXiv preprint arXiv:1205.2619*.

Vickson, Raymond G, Elkafi Hassini, and Nader Azad (2020). “A Benders decomposition approach to product location in carousel storage systems”. *Annals of Operations Research* 284.2, pp. 623–643.

Rokhforoz, Pegah and Olga Fink (2021). “Distributed joint dynamic maintenance and production scheduling in manufacturing systems: Framework based on model predictive control and Benders decomposition”. *Journal of Manufacturing Systems* 59, pp. 596–606.

Steimle, Lauren N, Vinayak S Ahluwalia, Charmee Kamdar, and Brian T Denton (2021a). “Decomposition methods for solving Markov decision processes with multiple models of the parameters”. *IISE Transactions*, pp. 1–58.

Warrington, Joseph, Paul N Beuchat, and John Lygeros (2019). “Generalized dual dynamic programming for infinite horizon problems in continuous state and action spaces”. *IEEE Transactions on Automatic Control* 64.12, pp. 5012–5023.

Warrington, Joseph (2019). “Learning continuous Q-functions using generalized Benders cuts”. *2019 18th European Control Conference (ECC)*. IEEE, pp. 530–535.

Puterman, Martin L and Moon Chirl Shin (1978). “Modified policy iteration algorithms for discounted Markov decision problems”. *Management Science* 24.11, pp. 1127–1137.

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement learning: An introduction*. MIT press.

Almasi, George S and Allan Gottlieb (1994). *Highly parallel computing.* Benjamin-Cummings Publishing Co., Inc.

Krishnamurthy, Vikram (2016). *Partially observed Markov decision processes.* Cambridge university press.

Alagoz, Oguzhan, Lisa M Maillart, Andrew J Schaefer, and Mark S Roberts (2007). "Determining the acceptance of cadaveric livers using an implicit model of the waiting list". *Operations Research* 55.1, pp. 24–36.

Shi, Yue, Yisha Xiang, and Mingyang Li (2019). "Optimal maintenance policies for multi-level preventive maintenance with complex effects". *IISE Transactions* 51.9, pp. 999–1011.

Asadi, Amin and Sarah Nurre Pinkley (2021). "A Monotone Approximate Dynamic Programming Approach for the Stochastic Scheduling, Allocation, and Inventory Replenishment Problem: Applications to Drone and Electric Vehicle Battery Swap Stations". *arXiv preprint arXiv:2105.07026.*

Zhuang, Weifen and Michael ZF Li (2012). "Monotone optimal control for a class of Markov decision processes". *European journal of operational research* 217.2, pp. 342–350.

Mattila, Robert, Cristian R Rojas, Vikram Krishnamurthy, and Bo Wahlberg (2017). "Computing monotone policies for Markov decision processes: a nearly-isotonic penalty approach". *IFAC-PapersOnLine* 50.1, pp. 8429–8434.

Lee, Ilbin, Marina A Epelman, H Edwin Romeijn, and Robert L Smith (2017). "Simplex algorithm for countable-state discounted Markov decision processes". *Operations Research* 65.4, pp. 1029–1042.

Braverman, Anton, Itai Gurvich, and Junfei Huang (2020). "On the Taylor expansion of value functions". *Operations Research* 68.2, pp. 631–654.

Chadès, Iadine, Guillaume Chapron, Marie-Josée Cros, Frédérick Garcia, and Régis Sabbadin (2014). "MDPtoolbox: a multi-platform toolbox to solve stochastic dynamic programming problems". *Ecography* 37.9, pp. 916–920.

The Cybersecurity and Infrastructure Security Agency (2021). *Critical Infrastructure Sectors.* URL: https://www.cisa.gov/critical-infrastructure-sectors.

Ouyang, Min (2014). “Review on modeling and simulation of interdependent critical infrastructure systems”. *Reliability Engineering and System Safety* 121, pp. 43–60.

U.S. Department of Energy (2021). *August 2003 Blackout.* URL: https://www.energy.gov/oe/services/electricity-policy-coordination-and-implementation/august-2003-blackout.

Yusta, Jose M., Gabriel J. Correa, and Roberto Lacal-Arántegui (2011). “Methodologies and applications for critical infrastructure protection: State-of-the-art”. *Energy Policy* 39, pp. 6100–6119.

Ouyang, Min (2017). “A mathematical framework to optimize resilience of interdependent critical infrastructure systems under spatially localized attacks”. *European Journal of Operational Research* 262, pp. 1072–1084.

Ghorbani-Renani, Nafiseh, Andrés D González, Kash Barker, and Nazanin Morshedlou (2020). “Protection-interdiction-restoration: Tri-level optimization for enhancing interdependent network resilience”. *Reliability Engineering & System Safety* 199, p. 106907.

Galbusera, Luca, Paolo Trucco, and Georgios Giannopoulos (2020). “Modeling interdependencies in multi-sectoral critical infrastructure systems: Evolving the DMCI approach”. *Reliability Engineering & System Safety* 203, p. 107072.

Ouyang, Min and Yiping Fang (2017). “A mathematical framework to optimize critical infrastructure resilience against intentional attacks”. *Computer-Aided Civil and Infrastructure Engineering* 32.11, pp. 909–929.

Fang, Yi-Ping and Enrico Zio (2019). “An adaptive robust framework for the optimization of the resilience of interdependent infrastructures under natural hazards”. *European Journal of Operational Research* 276.3, pp. 1119–1136.

Brown, Gerald, Matthew Carlyle, Javier Salmerón, and Kevin Wood (2006). “Defending Critical Infrastructure”. *INFORMS Journal on Applied Analytics* 36.6, pp. 530–544.

Baykal-Güersoy, Melike, Zhe Duan, H Vincent Poor, and Andrey Garnaev (2014). “Infrastructure security games”. *European Journal of Operational Research* 239.2, pp. 469–478.

Ferdowsi, Aidin, Anibal Sanjab, Walid Saad, and Narayan B Mandayam (2017). “Game theory for secure critical interdependent gas-power-water infrastructure”. *2017 Resilience Week (RWS).* IEEE, pp. 184–190.

Ma, Chris YT, David KY Yau, and Nageswara SV Rao (2013b). "Scalable solutions of Markov games for smart-grid infrastructure protection". *IEEE Transactions on Smart Grid* 4.1, pp. 47–55.

Ma, Chris Y. T., David K. Y. Yau, Xin Lou, and Nageswara S. V. Rao (2013a). "Markov Game Analysis for Attack-Defense of Power Networks Under Possible Misinformation". *IEEE Transactions on Power Systems* 28.2, pp. 1676–1686.

Kaplan, Edward H, Moshe Kress, and Roberto Szechtman (2010). "Confronting entrenched insurgents". *Operations Research* 58.2, pp. 329–341.

Zhuang, Jun, Vicki M Bier, and Oguzhan Alagoz (2010). "Modeling secrecy and deception in a multiple-period attacker–defender signaling game". *European Journal of Operational Research* 203.2, pp. 409–418.

Hausken, Kjell and Jun Zhuang (2011). "Governments' and terrorists' defense and attack in a T-period game". *Decision Analysis* 8.1, pp. 46–70.

Shan, Xiaojun and Jun Zhuang (2013). "Hybrid defensive resource allocations in the face of partially strategic attackers in a sequential defender–attacker game". *European Journal of Operational Research* 228.1, pp. 262–272.

Jose, Victor Richmond R and Jun Zhuang (2013). "Technology adoption, accumulation, and competition in multiperiod attacker-defender games". *Military Operations Research* 18.2, pp. 33–47.

Chang, Yanling, Alan L Erera, and Chelsea C White (2015). "A leader–follower partially observed, multiobjective Markov game". *Annals of Operations Research* 235.1, pp. 103–128.

Rass, Stefan and Quanyan Zhu (2016). "GADAPT: a sequential game-theoretic framework for designing defense-in-depth strategies against advanced persistent threats". *International conference on decision and game theory for security.* Springer, pp. 314–326.

Shan, Xiaojun and Jun Zhuang (2018). "Modeling cumulative defensive resource allocation against a strategic attacker in a multi-period multi-target sequential game". *Reliability Engineering & System Safety* 179, pp. 12–26.

Tullock, Gordon (2001). "Efficient rent seeking". *Efficient rent-seeking.* Springer, pp. 3–16.

Skaperdas, Stergios (1996). “Contest success functions”. *Economic theory* 7.2, pp. 283–290.

U.S. Department of Homeland Security (2022). *Homeland Infrastructure Foundation-Level Data*. URL: https://hifld-geoplatform.opendata.arcgis.com/.

Public Service Commission of Wisconsin (2021). *Underground Electric Transmission Lines*. Tech. rep. https://psc.wi.gov/Documents/Brochures/Under%20Ground%20Transmission.pdf.

Google (2022). *Google Maps Platform Distance Matrix API*. URL: https://developers.google.com/maps/documentation/distance-matrix/overview.

CareerExplorer (2022). *Security guard salary in Tennessee*. URL: https://www.careerexplorer.com/careers/security-guard/salary/tennessee/.

U.S. Census Bureau (2020). *Demographic Profile from 2020 Census, Knoxville city, Tennessee*. Tech. rep. https://data.census.gov/cedsci/profile?g=1600000US4740000.

Salman, Abdullahi M and Yue Li (2018). “A probabilistic framework for multi-hazard risk mitigation for electric power transmission systems subjected to seismic and hurricane hazards”. *Structure and Infrastructure Engineering* 14.11, pp. 1499–1519.

Conrad, Stephen H, Rene J LeClaire, Gerard P O’Reilly, and Huseyin Uzunalioglu (2006). “Critical national infrastructure reliability modeling and analysis”. *Bell Labs Technical Journal* 11.3, pp. 57–71.

Apenteng, Bettye A, Samuel T Opoku, Daniel Ansong, Emmanuel A Akowuah, and Evans Afriyie-Gyawu (2018). “The effect of power outages on in-facility mortality in healthcare facilities: evidence from Ghana”. *Global Public Health* 13.5, pp. 545–555.

Hirth, Richard A, Michael E Chernew, Edward Miller, A Mark Fendrick, and William G Weissert (2000). “Willingness to pay for a quality-adjusted life year: in search of a standard”. *Medical decision making* 20.3, pp. 332–342.

Murphy, Sherry L, Kenneth D Kochanek, Jiaquan Xu, and Elizabeth Arias (2021). “Mortality in the United States, 2020”. *NCHS Data Brief* 427. URL: https://www.cdc.gov/nchs/products/databriefs/db427.htm.

Sun, Ruirui, Zeynal Karaca, and Herbert S Wong (2018). *Trends in hospital inpatient stays by age and payer, 2000–2015: statistical brief# 235*. URL: https://www.hcup-us.ahrq.gov/reports/statbriefs/sb235-Inpatient-Stays-Age-Payer-Trends.jsp.

Centers for Disease Control and Prevention (2017). *Hospitals, beds, and occupancy rates, by type of ownership and size of hospital: United States, selected years 1975–2015*. URL: https://www.cdc.gov/nchs/data/hus/2017/089.pdf.

Tennessee Bureau of Investigation (2022). *CrimeInsight*. URL: https://www.tn.gov/tbi/divisions/cjis-division/tncrimeonline.html.

McCollister, Kathryn E, Michael T French, and Hai Fang (2010). "The cost of crime to society: New crime-specific estimates for policy and program evaluation". *Drug and alcohol dependence* 108.1-2, pp. 98–109.

Haynes, Hylton JG and Gary P Stein (2017). *US fire department profile 2015*. National Fire Protection Association Quincy, MA.

Federal Emergency Management Agency (2020). *FEMA benefit-cost analysis re-engineering (BCAR): Development of standard economic values version 6.0*. Tech. rep. https://www.fema.gov/sites/default/files/2020-08/fema_bca_toolkit_release-notes-july-2020.pdf.

Ericson, Sean and Lars Lisell (2020). "A flexible framework for modeling customer damage functions for power outages". *Energy Systems* 11.1, pp. 95–111.

U.S. Fire Administration (2022). *Tennessee Fire Loss and Fire Department Profile*. URL: https://www.usfa.fema.gov/data/statistics/states/tennessee.html#fatalities.

Appendix

Appendix A

MDP Benchmarking Problems

In the following, we introduce the four benchmarking problems from the literature. Specifically, we consider a queueing problem (de Farias and Van Roy 2003), an inventory management problem (Puterman 2014; Lee et al. 2017), a machine maintenance problem (Puterman 2014), and a data transmission problem (Krishnamurthy 2016). The equipment replacement problem and the data transmission problem are modified from their original form to allow arbitrary numbers of states and actions. For simplicity, key notation defined in each of the following problems, such as states, actions or certain parameters, may be repeatedly used.

A.1 The Queueing Problem

Consider a queue with N vacancies. Identical jobs are arriving in the queue at a rate p. Each job can be served using one of M services, with a probability $q \in \{q_1, \cdots, q_M\}$ of completing the job. At each decision epoch $t = 0, 1, \cdots, \infty$, the state of the system $s_t \in S := \{0, 1, \cdots, N\}$ is the number of jobs left in the queue, and the action $a_t \in A := \{1, 2, \cdots, M\}$ denotes which service to choose. The transition probability $T(s'|s, a)$ is defined based on the current state s. When $s = 0$,

$$T(s'|0,a) = \begin{cases} 1-p & \text{if } s' = 0, \\ p & \text{if } s' = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{A.1.1}$$

When $s = N$,

$$T(s'|N,a) = \begin{cases} q_a & \text{if } s' = N-1, \\ 1-q_a & \text{if } s' = N, \\ 0 & \text{otherwise.} \end{cases} \tag{A.1.2}$$

Finally, when $2 \leq s \leq N-1$,

$$T(s'|s,a) = \begin{cases} (1-p) \cdot q_a & \text{if } s' = s-1, \\ (1-p) \cdot (1-q_a) + p \cdot q_a & \text{if } s' = s, \\ p \cdot (1-q_a) & \text{if } s' = s+1, \\ 0 & \text{otherwise.} \end{cases} \tag{A.1.3}$$

The reward functions $R(s,a)$ is increasing in both s and a.

The problem is calibrated using parameters from the literature (de Farias and Van Roy 2003). The number of states and actions are specified by the experiments. We let $p = 0.2$ and draw the distribution of q uniformly from $[0,1]$, ensuring that $q_1 \leq q_2 \leq \cdots \leq q_M$. We use the reward function $R(s,a) = s + 60 \cdot a^3$.

A.2 The Inventory Management Problem

Consider an inventory of size N. At the beginning of the decision epoch t, the decision maker decides how many products to purchase and store in the inventory. Then, products in the inventory are used to satisfy customer demands at the end of the decision epoch. The state of the system $s_t \in S := \{0, 1, \cdots, N\}$ denotes the number of products in the inventory and the action $a_t \in A := \{0, 1, \cdots, N\}$ denotes the number of products to purchase. Note that at each decision epoch, a_t must satisfy $s_t + a_t \leq N$. Let $p(k)$, $k \in \{0, 1, \cdots, N\}$ denotes the probability of a demand of k products and $q(k) = \sum_{j=k}^{N} p(j)$ is thus the probability of a demand of at least k products. Then, the transition probability $T(s'|s, a)$ can be characterized as

$$T(s'|s,a) = \begin{cases} 0 & \text{if } s + a - s' < 0, \\ p(s + a - s') & \text{if } s + a - s' \geq 0 \text{ and } s' > 0, \\ q(s + a) & \text{otherwise.} \end{cases} \tag{A.2.1}$$

The reward is composed of several parts. Let b be the unit selling price of the product; K the fixed cost of ordering; c the unit cost of the product and h the unit holding cost of the product. The total reward is the selling revenue minus the ordering costs and the holding costs, i.e.,

$$R(s, a) = F(s + a) - O(a) - h \cdot (s + a), \tag{A.2.2}$$

where $F(s + a)$ is the expected revenue from selling the products,

$$F(s + a) = \sum_{j=0}^{s+a-1} b \cdot j \cdot p(j) + b \cdot (s + a) \cdot q(s + a), \tag{A.2.3}$$

$O(a)$ is the ordering cost,

$$O(a) = \begin{cases} 0 & \text{if a=0}, \\ K + c \cdot a & \text{otherwise}, \end{cases} \tag{A.2.4}$$

and $h \cdot (s + a)$ is the holding cost.

The problem is calibrated using parameters from the literature (Lee et al. 2017). The selling price b is uniformly sampled between 10 and 15; the fixed cost K is uniformly sampled between 3 and 5; the unit cost c is uniformly sampled between 5 and 7; the holding cost h is uniformly sampled between 0.1 and 0.2. The demand follows a Poisson distribution with expectation $\frac{1}{2}N$.

A.3 The Machine Maintenance Problem

Consider a machine with $N+1$ states. At each decision epoch t, the decision maker has the option to maintain the machine with M maintenance methods. Different maintenance methods show different effectiveness on the machine's operational condition, but also cost differently. The state of the system is $s_t \in S := \{0, 1, \cdots, N\}$, where 0 represents the best condition and N the worse condition. The action $a_t \in A := \{0, 1, \cdots, M\}$ represents which maintenance methods to use, where 0 suggests no maintenance. The cost and the effectiveness of maintenance increase with the action. At each epoch, the machine degrades from state s to s' with probability $p(s', s)$. Then, by taking an action a_t, the machine can be restored to condition 0 with probability $q(a)$. Thus, when $a_t = 0$, the transition probability $T(s'|s, 0) = p(s', s)$. When $a_t = 1, 2, \cdot, M$, the transition probability

$$T(s'|s,a) = \begin{cases} q(a) \cdot p(s', 0) & \text{if } s' < s, \\ q(a) \cdot p(s', 0) + (1 - q(a)) \cdot p(s', s) & \text{otherwise.} \end{cases} \tag{A.3.1}$$

The reward is calculated as the machine's fixed reward C_r minus the operational cost $C_o(s)$ and the maintenance cost $C_m(a)$, i.e., $R(s, a) = C_r - C_o(s) - C_m(a)$. The operational cost $C_o(s)$ is increasing in s and the maintenance cost $C_m(a)$ is increasing in a.

The problem is calibrated as follows. The degradation probability $p(s', s)$ is uniformly sampled with the increasing failure rate property. The restoration probability $q(a)$ is uniformly sampled with $q(0) = 0$ and the ordering $q(1) \leq q(2) \leq \cdots q(M)$. We let the fixed reward $C_r = \frac{1}{2}N$. The operational cost $C_o(s)$ is uniformly drawn between 0 and $\frac{3}{4}N$, with ordering that $C_o(0) \leq C_o(1) \leq \cdots C_o(N)$. The maintenance cost $C_m(a) = \frac{1}{10}N \cdot a$.

A.4 The Data Transmission Problem

Consider a transmission channels with N conditions and M packages to transmit. The decision maker has the option to choose from K transmission options, or does not transmit at all. The transmission success rate is positively correlated to the channel condition and transmission option. Condition of the channel chances independent of the transmission actions. The goal is to transmit all packages as soon as possible. The state of the system $s_t := (a, b)$ is the combination of channel condition $a = 1, 2, \cdots, N$ and the number of packages $b = 0, 1, \cdots, M$. Thus, there are in total $N \cdot (M + 1)$ states. The action $a_t \in A := \{0, 1, \cdots, K\}$ represents which transmission option to use, where 0 represents not to transmit the package. Each transmission option corresponds to a success rate $p(s(0), a)$ under the state s, where $p(s(0), a)$ is increasing in both $s(0)$ and a. The channel condition transitions from $s(0)$ to $s'(0)$ independent of the transmission options, with probability $q(s'(0), s(0))$. Thus, when $a = 0$, the transition probability

$$T(s'|s, 0) = \begin{cases} q(s'(0), s(0)) & \text{if } s(1) = s'(1), \\ 0 & \text{otherwise.} \end{cases} \tag{A.4.1}$$

When $a \geq 1$, the transition probability

$$T(s'|s, a) = \begin{cases} q(s'(0), s(0)) & \text{if } s(1) = s'(1) = 0, \\ (1 - p(s(0), a)) \cdot q(s'(0), s(0)) & \text{if } s(1) = s'(1) \neq 0, \\ p(s(0), a) \cdot q(s'(0), s(0)) & \text{if } s(1) - 1 = s'(1), \\ 0 & \text{otherwise.} \end{cases} \tag{A.4.2}$$

The reward function consists of two parts: the package holding cost $C_h(s(1)) \geq 0$ and the transmission cost $C_t(a) \geq 0$, where $C_h(s(1))$ is increasing in $s(1)$ and $C_t(a)$ is increasing in a. The reward $R(s,a) = -C_h(s(1))$ if $a = 0$ and $R(s,a) = -C_h(s(1)) - C_t(a)$ otherwise.

The problem is calibrated as follows. The transmission success rate $p(s(0), a)$ is uniformly drawn with a constraint to be increasing in both $s(0)$ and a. The channel condition transition $q(s'(0), s(0))$ is uniformly drawn. We let $C_h(s(1)) = c_h \cdot s(1)$, where c_h is uniformly sampled between 0 and 5. We let $C_t(0) = 0$ and uniformly sample $C_t(a)$ between 5 and 15, with an ordering $C_t(1) \leq C(2) \leq \cdots \leq C_t(K)$.

Appendix B

LSSD Benchmarking Problems

In the following, we extend the four benchmarking problems described in Appendix A for the SSSD framework.

B.1 The Queueing Problem

As discussed, in `queue`, we consider a single server, with m types of service rates. Customers arrive at the rate p. Here, the strategic decision is to establish a "waiting area" before the service begins, i.e., decide on the capacity of the queue. A larger queue capacity means that more customers can be served, but is associated with higher costs. A shorter queue reduces the cost, but also faces potential losses of demand when there is no vacancy in the queue.

In the following formulation, we let x, $0 \leq x \leq n$ be the queue capacity, where n is the maximum queue capacity, representing an upper bound on x. The operational decisions in the second stage are the service rate at each time $t = 0, 1, \ldots, \infty$. The second stage variable $y_{s,a} \geq 0$ denotes how many times a service a is used under the state (queue length) s. The objective of the model is to minimize the total cost. For consistency, we still use a max objective:

$$\max \quad -c \cdot x + \sum_{s \in S} \sum_{a \in A} r_{s,a} y_{s,a} \tag{B.1.1}$$

(B.1.2)

In order to define the transition probability and the reward, we require binary auxiliary variables to decide the relationship between x and s. We let $\zeta_s = 1$ if $s < x$, and 0 otherwise; $\delta_s = 1$ if $s = x$, and 0 otherwise; $\sigma_s = 1$ if $s > x$, and 0 otherwise. The constraints between x and the auxiliary variables can be defined as follows

$$M \cdot \zeta_s \geq (x - \frac{1}{2}) - s \quad \forall\ s \in S; \tag{B.1.3}$$

$$M \cdot (\zeta_s - 1) \leq (x - \frac{1}{2}) - s \quad \forall\ s \in S; \tag{B.1.4}$$

$$M \cdot \sigma_s \geq s - (x + \frac{1}{2}) \quad \forall\ s \in S; \tag{B.1.5}$$

$$M \cdot (\sigma_s - 1) \leq s - (x + \frac{1}{2}) \quad \forall\ s \in S; \tag{B.1.6}$$

$$M \cdot (\delta_s - 1) \leq s - (x - \frac{1}{2}) \quad \forall\ s \in S; \tag{B.1.7}$$

$$M \cdot (\delta_s - 1) \leq (x + \frac{1}{2}) \quad \forall\ s \in S; \tag{B.1.8}$$

$$\sum_{s \in S} \delta_s = 1. \tag{B.1.9}$$

To model the loss of demand due to small queue capacity, we define a parameter $\psi = 2000$, representing the economic loss of losing a customer. Using the auxiliary variables, the constraints regarding $\boldsymbol{\tau}$ and $\boldsymbol{r}$ can be modeled as follows

$$\tau_{s',s,a} = 1 - p \quad \forall\ s = 0, s' = 0, s, s' \in S, a \in A; \tag{B.1.10}$$

$$\tau_{s',s,a} = p \quad \forall\ s = 0, s' = 1, s, s' \in S, a \in A; \tag{B.1.11}$$

$$\tau_{s',s,a} = \delta_s \cdot q_a + \zeta_s(1 - p)q_a \quad \forall\ s \geq 1, s' = s - 1, s, s' \in S, a \in A; \tag{B.1.12}$$

$$\tau_{s',s,a} = \delta_s \cdot (1 - q_a) + \zeta_s[(1 - p)(1 - q_a) + pq_a] + \sigma_s \quad \forall\ s \geq 1, s' = s, s, s' \in S, a \in A; \tag{B.1.13}$$

$$\tau_{s',s,a} = \zeta_s \cdot p \cdot (1 - q_a) \quad \forall\ s \geq 1, s' = s + 1, s, s' \in S, a \in A; \tag{B.1.14}$$

$$r_{s,a} = (1 - \sigma_s)[-(s + 60 \cdot q_a^3) - \psi \cdot \delta_s] \quad \forall\ s = 0, s \in S, a \in A; \tag{B.1.15}$$

$$r_{s,a} = (1 - \sigma_s)[-(s + 60 \cdot q_a^3) - \psi \cdot (1 - q_a) \cdot p \cdot \delta_s] \quad \forall\ s \neq 0, s \in S, a \in A. \tag{B.1.16}$$

Lastly, the operational decisions are solved using the MDP constraint

$$\sum_{a \in A} y_{s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} \tau_{s',s,a} y_{s,a} = \alpha_s \quad \forall\ s \in S. \tag{B.1.17}$$

B.2 The Inventory Management Problem

In `inventory`, the strategic decision is to decide the optimal inventory capacity, so that future customer demand can be satisfied. We use $x \in \mathbb{N}_+$, $0 \leq x \leq N$ to denote the inventory capacity, where N is the maximum inventory capacity. Other parameters of the model follow what was shown in Appendix A.2. We let β be the cost of unit inventory space. The objective of the problem can be written as

$$\max \quad -\beta x + \sum_{s \in S} \sum_{a \in A} r_{s,a} y_{s,a}. \tag{B.2.1}$$

To formulate the constraints for the inventory problem, we first introduce the following auxiliary variables:

- $\delta_{s,a} \in \{0,1\}$, $\forall\ s \in S, a \in A$: whether $s + a > x$;
- $\sigma_s \in \{0,1\}$, $\forall\ s \in S$: whether $s > x$;
- $\zeta_{s,s'} \in \{0,1\}$, $\forall\ s, s' \in S$: whether $s' = x - s$;
- $\omega_s \in \{0,1\}$, $\forall\ s \in S$: whether $s = x$.

Due to the complexity of the transition probability and the reward of `inventory`, we further define $\lambda^i_{s',s,a} \in \{0,1\}$, $\forall\ i = 1, \ldots, 7, s, s' \in S, a \in A$ and $\mu^j_{s,a,s'} \in \{0,1\}$, $\forall\ j = 1,2,3, s, s' \in S, a \in A$ to help distinguish different scenarios. For the transition probability, $\boldsymbol{\lambda}$ describes seven scenarios. Note that non-negative customer demand ensures that $s + a < s'$, on which the following scenarios are based:

1. $\lambda^1_{s',s,a} = 1$ if $s > x, s' = s$;
2. $\lambda^2_{s',s,a} = 1$ if $s > x, s' \neq s$;
3. $\lambda^3_{s',s,a} = 1$ if $s \leq x, s' > x$;
4. $\lambda^4_{s',s,a} = 1$ if $s \leq x, 0 < s' \leq x, s + a \leq x$;
5. $\lambda^5_{s',s,a} = 1$ if $s \leq x, 0 < s' \leq x, s + a > x$;
6. $\lambda^6_{s',s,a} = 1$ if $s \leq x, s' = 0, s + a \leq x$;
7. $\lambda^7_{s',s,a} = 1$ if $s \leq x, s' = 0, s + a > x$.

For the rewards, the following three conditions are distinguished using the variable $\boldsymbol{\mu}$:

1. $\mu^1_{s,a,s'} = 1$, if $s' \leq x$;
2. $\mu^2_{s,a,s'} = 1$, if $s' > x, s + a > x$;
3. $\mu^3_{s,a,s'} = 1$, if $s' > x, s + a \leq x$.

Now, we present all the constraints of `inventory`. First, we show constraints associated with variables $\boldsymbol{\delta}$, $\boldsymbol{\sigma}$, $\boldsymbol{\zeta}$, and $\boldsymbol{\omega}$:

$$M\delta_{s,a} \geq s + a - (x + \frac{1}{2}) \quad \forall\ s \in S, a \in A; \tag{B.2.2}$$

$$M(\delta_{s,a} - 1) \leq s + a - (x + \frac{1}{2}) \quad \forall\ s \in S, a \in A; \tag{B.2.3}$$

$$M\sigma_s \geq s - (x + \frac{1}{2}) \quad \forall\ s \in S; \tag{B.2.4}$$

$$M(\sigma_s - 1) \leq s - (x + \frac{1}{2}) \quad \forall\ s \in S; \tag{B.2.5}$$

$$\sigma_s + \sum_{s'} \zeta_{s,s'} \geq 1 \quad \forall\ s \in S; \tag{B.2.6}$$

$$M(\zeta_{s,s'} - 1) \leq (x + \frac{1}{2}) - (s + s') \quad \forall\ s \in S; \tag{B.2.7}$$

$$M(\zeta_{s,s'} - 1) \leq (s + s') - (x - \frac{1}{2}) \quad \forall\ s \in S; \tag{B.2.8}$$

$$\sum_{s \in S} \omega_s = 1; \tag{B.2.9}$$

$$M(\omega_s - 1) \leq s - (x - \frac{1}{2}) \quad \forall\ s \in S; \tag{B.2.10}$$

$$M(\omega_s - 1) \leq (x + \frac{1}{2}) - s \quad \forall\ s \in S. \tag{B.2.11}$$

Next, we show constraints used to calculate $\boldsymbol{\lambda}$:

$$\lambda^1_{s',s,a} = \sigma_s \quad \forall\ s, s' \in S, s' = s, a \in A; \tag{B.2.12}$$

$$\lambda^1_{s',s,a} = 0 \quad \forall\ s, s' \in S, s' \neq s, a \in A; \tag{B.2.13}$$

$$\lambda^2_{s',s,a} = 0 \quad \forall\ s, s' \in S, s' = s, a \in A; \tag{B.2.14}$$

$$\lambda^2_{s',s,a} = \sigma_s \quad \forall\ s, s' \in S, s' \neq s, a \in A; \tag{B.2.15}$$

$$\lambda^3_{s',s,a} \geq (\sigma_{s'} + (1 - \sigma_s)) - \frac{3}{2} \quad \forall\ s, s' \in S, a \in A; \tag{B.2.16}$$

$$\lambda^3_{s',s,a} \leq \frac{1}{2} \cdot (\sigma_{s'} + (1 - \sigma_s)) \quad \forall\ s, s' \in S, a \in A; \tag{B.2.17}$$

$$\lambda^4_{s',s,a} = 0 \quad \forall\ s, \in S, s' = 0, a \in A; \tag{B.2.18}$$

$$\lambda^4_{s',s,a} \geq [(1 - \sigma_{s'}) + (1 - \sigma_s) + (1 - \delta_{s,a})] - \frac{5}{2} \quad \forall\ s, s' \in S, s' \neq 0, a \in A; \tag{B.2.19}$$

$$\lambda^4_{s',s,a} \leq \frac{1}{3} \cdot [(1 - \sigma_{s'}) + (1 - \sigma_s) + (1 - \delta_{s,a})] \quad \forall\ s, s' \in S, s' \neq 0, a \in A; \tag{B.2.20}$$

$$\lambda^5_{s',s,a} = 0 \quad \forall\ s, \in S, s' = 0, a \in A; \tag{B.2.21}$$

$$\lambda^5_{s',s,a} \geq [(1 - \sigma_{s'}) + (1 - \sigma_s) + \delta_{s,a}] - \frac{5}{2} \quad \forall\ s, s' \in S, s' \neq 0, a \in A; \tag{B.2.22}$$

$$\lambda^5_{s',s,a} \leq \frac{1}{3} \cdot [(1 - \sigma_{s'}) + (1 - \sigma_s) + \delta_{s,a}] \quad \forall\ s, s' \in S, s' \neq 0, a \in A; \tag{B.2.23}$$

$$\lambda^6_{s',s,a} = 0 \quad \forall\ s, \in S, s' \neq 0, a \in A; \tag{B.2.24}$$

$$\lambda^6_{s',s,a} \geq [(1 - \sigma_{s'}) + (1 - \sigma_s) + (1 - \delta_{s,a})] - \frac{5}{2} \quad \forall\ s, s' \in S, s' = 0, a \in A; \tag{B.2.25}$$

$$\lambda^6_{s',s,a} \leq \frac{1}{3} \cdot [(1 - \sigma_{s'}) + (1 - \sigma_s) + (1 - \delta_{s,a})] \quad \forall\ s, s' \in S, s' = 0, a \in A; \tag{B.2.26}$$

$$\lambda^7_{s',s,a} = 0 \quad \forall\ s, \in S, s' \neq 0, a \in A; \tag{B.2.27}$$

$$\lambda^7_{s',s,a} \geq [(1 - \sigma_{s'}) + (1 - \sigma_s) + \delta_{s,a}] - \frac{5}{2} \quad \forall\ s, s' \in S, s' = 0, a \in A; \tag{B.2.28}$$

$$\lambda^7_{s',s,a} \leq \frac{1}{3} \cdot [(1 - \sigma_{s'}) + (1 - \sigma_s) + \delta_{s,a}] \quad \forall\ s, s' \in S, s' = 0, a \in A. \tag{B.2.29}$$

The transition probability can then be calculated using the following constraints:

$$\tau_{s',s,a} = 0 \quad \forall\ s, s' \in S, a \in A, s + a - s' < 0; \tag{B.2.30}$$

$$\begin{aligned}\tau_{s',s,a} = {} & \lambda^1_{s',s,a} \cdot 1 + \lambda^2_{s',s,a} \cdot 0 + \lambda^3_{s',s,a} \cdot 0 + \lambda^4_{s',s,a} \cdot p(s + a - s') + \lambda^5_{s',s,a} \cdot \sum_{\bar{s} \in S} \zeta_{s',\bar{s}} \cdot p(\bar{s}) \\ & + \lambda^6_{s',s,a} \cdot q(s + a) + \lambda^7_{s',s,a} \cdot \sum_{\bar{s} \in S} \omega_{\bar{s}} \cdot q(\bar{s}) \quad \forall\ s, s' \in S, a \in A, s + a - s' \geq 0;\end{aligned} \tag{B.2.31}$$

$$\sum_{s'} \tau_{s',s,a} = 1 \quad \forall\ s \in S, a \in A. \tag{B.2.32}$$

Similarly, the following constraints calculate the values of $\boldsymbol{\mu}$ and $\boldsymbol{r}$:

$$\mu^1_{s,a,s'} = 1 - \sigma_{s'} \quad \forall\ s, s' \in S, a \in A; \tag{B.2.33}$$

$$\mu^2_{s,a,s'} \geq (\sigma_{s'} + \delta_{s,a}) - \frac{3}{2} \quad \forall\ s, s' \in S, a \in A; \tag{B.2.34}$$

$$\mu^2_{s,a,s'} \geq \frac{1}{2} \cdot (\sigma_{s'} + \delta_{s,a}) \quad \forall\ s, s' \in S, a \in A; \tag{B.2.35}$$

$$\mu^3_{s,a,s'} \geq [\sigma_{s'} + (1 - \delta_{s,a})] - \frac{3}{2} \quad \forall\ s, s' \in S, a \in A; \tag{B.2.36}$$

$$\mu^3_{s,a,s'} \geq \frac{1}{2} \cdot [\sigma_{s'} + (1 - \delta_{s,a})] \quad \forall\ s, s' \in S, a \in A; \tag{B.2.37}$$

$$\begin{aligned}r_{s,a} = {} & \sum_{s' \in S} \Big[\mu^1_{s,a,s'} \cdot b \cdot s' \cdot p(s') + \mu^2_{s,a,s'} \cdot b \cdot x \cdot p(s') + \mu^3_{s,a,s'} \cdot b \cdot (s + a) \cdot p(s + a)\Big] \\ & - \begin{cases} K + c \cdot a & a \neq 0 \\ 0 & a = 0 \end{cases} - h(s + a)(1 - \delta_{s,a}) - h \cdot x \cdot \delta_{s,a} \quad \forall\ s \in S, a \in A.\end{aligned} \tag{B.2.38}$$

Finally, the operational decisions are solved using the MDP constraints

$$\sum_{a \in A} y_{s,a} - \gamma \sum_{a \in A} \sum_{s' \in S} \tau_{s',s,a} y_{s,a} = \alpha_s \quad \forall\ s \in S. \tag{B.2.39}$$

Vita

Zeyu Liu was born on February 27, 1996, in Nanjing, China. He attended Chaha'er Road Elementary School from six to twelve, where he played on the school's baseball team and won first place in a city-wide competition with his team. Zeyu was later admitted to Nanjing No.9 Middle school and participated in many student service and administration programs. After three years of study, Zeyu entered Nanjing No. 13 High School, located by the side of the glamorous Xuanwu Lake, at the foot of the ancient city walls and the emerald Zijin Mountain. High school life was stressful yet versatile. To everyone's surprise, including himself, Zeyu performed extraordinarily in the National College Entrance Examination in 2014. He received and accepted the offer from the School of Economics and Management, at Southeast University, descendent of one of the oldest, and most renowned universities in China. In college, Zeyu studied and had fun, served student communities and joined hobby clubs, fell in love and formed bonds with friends. He also learned the basics of scientific research from Dr. Jia Shu, who provided valuable counseling for his future path. After graduation from college, Zeyu was admitted to the University of Tennessee, Knoxville in 2018 as a Ph.D. student, under the advisement of Dr. Xueping Li and Dr. Anahita Khojandi. Thanks to the guidance from his advisors, Zeyu published his first conference paper in 2019, and his first journal paper in 2020. Besides research, Zeyu is also an active gamer who enjoys a variety of video games and board games, a model builder who assembles and paints models including military warships and Warhammer miniatures, a guitarist, a movie-lover, and an inline skater. In 2022, Zeyu was hired by the Department of Industrial and Management Systems Engineering, at West Virginia University to become a member of the faculty. His parents are proud of him.