June 2022

Computational Methods for Solving the Combinatorial Optimization Problems in Transportation

Xufei Liu
University of South Florida

Computational Methods for Solving the Combinatorial Optimization Problems in Transportation

by

Xufei Liu

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Industrial and Management Systems Engineering
College of Engineering
University of South Florida

Major Professor: Changhyun Kwon, Ph.D.
Ankit Shah, Ph.D.
Hadi Charkhgard, Ph.D.
He Zhang, Ph.D.
Xiaopeng Li, Ph.D.

Date of Approval:
June 5, 2022

Keywords: Vehicle Routing, Network Design, Car Sharing, Cutting Plane, Metaheuristic

Dedication

This dissertation is dedicated to my parents Yingge Liu and Lihong Xu, and my dear grandparents for their endless love and support.

Acknowledgments

There are many people that I would like to express my heartfelt thanks to. First of all, I would like to express my greatest gratitude to my advisor Dr. Changhyun Kwon for his guidance and patience throughout my PhD. I am also grateful to my committee members, Dr. Changhyun Kwon, Dr. Ankit Shah, Dr. Hadi Charkhgard, Dr. He Zhang and Dr. Xiaopeng Li for their advice and guidance through my doctoral study.

Table of Contents

List of Tables

List of Figures

Abstract

This dissertation discusses three transportation problems. The first problem is a bi-level optimization problem that simultaneously optimizes facility locations and network design in hazardous materials transportation. In the upper level, the leader intends to reduce the facility setup cost and the hazmat exposure risk, by choosing facility locations and road segments to close for hazmat transportation. When making such decisions, the leader anticipates the response of the followers who want to minimize the transportation costs. A robust optimization approach with multiplicative uncertain parameters and polyhedral uncertainty sets is applied to deal with the uncertain risk and demand.

The second problem comes from the Free-floating electric vehicle sharing systems. It allows users to pick up and return an electric vehicle at any permissible parking location within a service area. Such service flexibility can drive a severe spatial imbalance between vehicle availability and trip demands. We consider the operations to relocate the EV fleet to meet the next day's demand with sufficient battery levels. This relocation operation involves a complicated routing problem for a fleet of shuttles to transport the staff drivers who relocate the EVs to proper demand locations. We devise an efficient algorithm, which adapts the Adaptive Large Neighborhood Search framework. The experimental results validate the efficiency and effectiveness of our proposed algorithm and prove it is quite flexible to adapt to a dynamic environment.

The third problem is arc routing problem with the truck and the drones which cooperatively service the required edges. While the trucks follow road networks, drones can fly directly between any two points and off the network. The cooperation of the truck and the drone extends the traditional arc routing problem. We consider routing the truck and the drone with the limited flight range. An Adaptive Large Neighborhood Search is devised to

solve the Drone-Truck Arc Routing Problem. The experimental results on the small-size and large-size instances validate the efficiency and effectiveness of the proposed method.

Chapter 1: Introduction

For the combinatorial problems in transportation, vehicle routing aims to find an optimal route that wants the minimum total cost, the minimum risk, or the minimum completion time. The routing is mainly decision in transportation problems. Three problems are considered in the dissertation: the combined facility location and network design problem in hazardous materials transportation, rebalancing free-floating electric vehicle sharing systems, and Drone-Truck arc routing problem.

The first problem is a leader-follower decision problem in the form of bi-level optimization. In the upper level, the leader aims to minimize the total facility construction costs and hazmat exposure risks by determining facility locations and available roads for hazmat transportation. The leader affects the followers who intend to minimize their transportation costs when designing the road network. We apply a robust optimization approach with multiplicative uncertain parameters and polyhedral uncertainty sets to deal with the uncertainty in the exposure risk and the demand. A bi-level integer programming model is formulated where the upper level is a min-max problem and the lower level is a shortest-path problem. We devise an exact algorithm that combines a cutting plane algorithm with Benders decomposition and derive a single-level reformulation. Comparisons between two approaches are made on the Ravenna city data, in terms of objectives and the running time. The analysis on small and large size instances demonstrates that the proposed cutting plane algorithm performs much better than Gurobi as the problem size increases. The proposed cutting plane algorithm is an effective exact method for solving the robust combined facility location-network design problem.

The second problem considers the EV relocation and shuttle routing for the rebalancing operation of free-floating EV sharing systems (FFEVSS). One of the key operational decisions for the carsharing company is how to relocate the EV fleet to meet the next day's demand with sufficient battery levels. We develop a metaheuristic based on an adaptive large neighborhood search for this problem that determines where to relocate each EV and how to route the shuttles that transport the staff drivers synchronously. We apply our method to conduct numerical experiments using both randomly generated data and actual FFEVSS data in Amsterdam. We found that ALNS outperforms EBNSM both in the solution quality and the computational time. ALNS also produces better solutions than the RL approach but requires much longer computational time than RL. The experiments reveal that providing the RL solution as the initial solution for ALNS is an effective and efficient solution strategy that can take advantage of both approaches, achieving the best solution quality and reducing the computational time significantly. We also demonstrate how ALNS can be modified to solve the problem where staff drivers carry a personal mobility vehicle such as a scooter. The further analysis provides practical recommendations on which mode of transportation will be more efficient—i.e., a small number of shuttles with large capacity or a large number of shuttles with small capacity (or even personal mobility)—in terms of total operational cost as well as wait times. Lastly, we show that our ALNS is quite flexible to be applied to a dynamic environment when it destroys an incumbent solution partially and repairs to a new solution in each iteration. Specifically, our numerical results highlight the usefulness of our flexible ALNS method for an environment where some EV demands are removed or added in the course of EV relocation operations.

The third problem is Drone-Truck arc routing problem. Arc routing problems are widely used in many fields, including traffic monitoring, infrastructure inspection, and security. The drone and the truck cooperatively service all required edges at least once. Since the drone can fly off the road network, the DT-ARP extends the traditional ARP. With a limited battery capacity, the drone needs to fly from and to vehicles for a replacement of battery.

The key challenge is how to determine the truck and drone routes to minimize the completion time. In order to get the optimal solution, we transform the ARP into VRP with two kinds of rules and formulate a mixed-integer programming. The experiments reveal that MIP formulation can solve the problem well for the small-size network. However, for a large-size network, an efficient and effective metaheuristic is necessary. A metaheuristic method based on Adaptive Large Neighborhood Search (ALNS) is proposed to solve the Drone-Truck Arc Routing Problem. The effectiveness of ALNS is evaluated over the small-size randomly generated instances and large-size undirected rural postman problem instances. The experimental results show the advantage of ALNS in the solution quality and run time for two cases: One Truck-One Drone and One Truck-Two Drones. Further analysis on the truck/drone speed and the drone's maximum flight range reveals the difficulty to solve the problem. The robustness of ALNS is also discussed and evaluated by the standard deviation of multiple repeated solutions on the same instance.

In summary, the goal of this dissertation is to answer the following questions:

- How to design the network and choose locations to construct facilities by the administrator decisions and to choose the transportation routes by the truck drivers in the hazmat material transportation?
- How to route shuttles to distribute staff drivers to relocate EVs for rebalancing the free-floating EV sharing systems?
- How to efficiently route drone and truck to service all required edges cooperatively with the aim of minimizing the completion time?

The dissertation can be summarized as follows: In Chapter 2, we propose an exact method that combines the cutting plane algorithm with Benders Decomposition for the combined facility location and network design problem. Chapter 3 proposes a metaheuristic method, named Adaptive Large Neighborhood Search, to relocate EVs and route shuttles. In Chapter

4, an adaptive large neighborhood search method is devised to solve Drone-Truck arc routing problem. Chapter 5 summarizes three problems.

Chapter 2: Exact Robust Solutions for the Combined Facility Location and Network Design Problem in Hazardous Materials Transportation

Portions of this chapter have been previously published in IISE Transactions (2020), 52(10), 1156-1172. The copyright permissions for the reuse of previously published material in this chapter can be found in Appendix A.

2.1 Introduction

Hazardous materials (hazmat) are "solids, liquids, or gases that are harmful to people, property, and the environment" (United Nations, 2009). A large amount of hazmat is generated in industrial production and transported over various transportation modes. Trucks are the most popular mode of transporting hazmat (Erkut et al., 2007). For example, in the U.S., more than 2.4 billion tons of hazmat were transported by trucks in 2012 (U.S. Department of Transportation, 2015). Accidents involving hazmat can create catastrophic consequences; hence the road system is facing pressure on the constantly increasing amount of hazmat shipments. Managing risk in hazmat transportation is important in any industrial society.

In most cases, the hazmat producers are responsible to carry hazmat to an appropriate processing facility. The hazmat carriers make their choices about the transportation route, usually, aiming to minimize the shipment cost. The local route decision of hazmat carriers is beyond the control of the government, who considers the impact of hazmat transportation from a global perspective of managing the entire road network and other infrastructure systems. The government wants to minimize the total shipment exposure risk and total facility construction costs. To achieve this goal, the government may consider road-ban

policies to specify the available and unavailable roads for hazmat shipments. Such policies prohibit hazmat carriers from choosing a route with small transportation costs but with great hazmat exposure risk. The problem to determine such road-ban policies is called a hazmat network design problem in the literature.

In this chapter, we consider a *combined* hazmat facility locations and network design problem. We assume that origin points where hazardous materials are produced are known, but destination points (disposal facility location) are not. Instead, hazmat carriers are assumed to choose the nearest facility if multiple facilities are available within the network; therefore, the route decision of hazmat carriers is dependent on the location decision of the government. When the government determines the locations of hazmat processing facilities, we assume that the government also considers a road-ban policy to design the hazmat network, upon which the route decision of hazmat carriers also depends. This structure of hierarchical decision-making has been considered in a bi-level optimization framework in the literature (Kara and Verter, 2004; Erkut and Alp, 2007; Gzara, 2013; Berglund and Kwon, 2014; Marcotte et al., 2009; Sun et al., 2015). We will present our problem as a bi-level optimization problem as well.

We consider uncertain hazmat transportation demands and uncertain hazmat accident risks. By assuming data for the demands and risks are available as intervals, we consider the worst-case scenario using a robust optimization approach. We will consider polyhedral uncertainty sets as considered in Bertsimas and Sim (2003). In our problem, the two uncertain parameters form a product in the objective function, for which we adopt the approach of Kwon et al. (2013).

Our work is closely related to Berglund and Kwon (2014) and Gzara (2013). Berglund and Kwon (2014) have considered a robust hazmat facility location problem. Our modeling approach for the robust combined facility location and network design problem extends the work of Berglund and Kwon (2014). The computational method proposed by Berglund and Kwon (2014), however, is a genetic algorithm, which does not produce an exact optimal

solution in general. In this chapter, for the combined problem, we devise an exact algorithm by adopting the cutting plane algorithm of Gzara (2013) and combining with Benders decomposition.

Gzara (2013) has devised a cutting plane algorithm for solving the bi-level hazmat network design problem. The model of Gzara (2013), however, only considered a network design decision without considering data uncertainty. Our problem is a robust optimization problem that considers the facility location decision and the network design decision jointly. As we adopt the cutting plane algorithm of Gzara (2013) to the robust combined problem, we have revised the cut generation method for the joint decision. We also simplify the inequalities in the cuts and eliminate the need for additional binary variables. In our problem, the master problem is significantly harder to solve, mainly due to the robustness consideration; we devise a Benders decomposition (Benders, 1962) approach for solving the master problem. While a Benders decomposition approach has been used to solve a single-level reformulation of the deterministic hazmat network design problem (Fontaine and Minner, 2018), we use Benders decomposition to solve the robust master problem involving uncertainty within the cutting plane algorithm framework for the joint decision of facility location and network design.

The contributions of this chapter are summarized as follows. We consider a combined facility location and network design problem for hazmat transportation. By assuming data uncertainty, we formulate a robust optimization problem as a bi-level mixed-integer optimization problem, where the upper-level problem has a min-max structure. We propose a cutting plane algorithm incorporated with Benders decomposition to solve the robust combined problem.

The remainder of this chapter is as follows. In Section 2.2, more related works are summarized and the relevance to our work is discussed. In Section 2.3, a bi-level location-network design mathematical optimization model is formulated. In Section 2.4, we present a cutting plane algorithm, combined with Benders decomposition, to solve the optimization

problem. In Section 2.5, we provide a single-level reformulation of the bi-level robust problem. Results from numerical experiments are discussed in Section 2.6. Finally, conclusions and future researches are provided in Section 2.7.

2.2 Literature Review

In this section, we review the literature in the four categories: hazmat facility location, hazmat network design, combined facility and network design in non-hazmat context, and robust optimization approaches in hazmat transportation.

2.2.1 Hazmat Facility Location Problems

There are a variety of methods for facility location problems in hazmat transportation. The related studies assume that facility locations are not given and need to solve a routing problem. Carotenuto et al. (2007) propose two greedy algorithms to select the path which minimizes the total risk. Xie et al. (2012) study multi-objective hazmat model that optimizes facility locations and routes in the long-distance transportation and solve the mixed integer linear program by CPLEX. Jarboui et al. (2013) propose various neighborhood search (VNS) heuristics for solving location-routing problem. Samanlioglu (2013) studies a location-routing problem and proposes a lexicographic weighted Tchebycheff formulation to minimize multi-objectives of total cost, transportation risk, and site risk. Ardjmand et al. (2015) apply a novel genetic algorithm for location-routing problem in facilities and disposal sites. Romero et al. (2016) analyze location-routing decisions considering equity based on Gini coefficient and propose a method that combines Lagrangian relaxation with column generation. Rabbani et al. (2018) emphasize on hazmat formulation restriction, i.e., incompatibility between different kinds of waste with multi-objectives of minimizing total cost, transportation risk, and site risk. They use Nondominated Sorting Genetic Algorithm (NSGA-II) and Multi-Objective Particle Swarm Optimization (MOPSO) to solve the problem. For earlier works, see Berglund and Kwon (2014) and references therein.

2.2.2 Hazmat Network Design Problems

There are also some research papers related to network-design problem. The routing is also considered when the locations of origin-destination pairs are given. Verter and Kara (2008) provide a path-based formulation for network design hazmat shipment problem and compromise between exposure risk and economic viability. Garrido (2008) and Marcotte et al. (2009) study a network-design problem where origin-destination pairs are given and aim to minimize exposure risk. They design the network by road pricing method, and Wang et al. (2012) improve the method and propose a dual-toll pricing policy. Bianco et al. (2009) provide a linear bi-level programming formulation for the hazmat transportation network design that considers minimizing total risk and risk equity. They propose a heuristic algorithm to find a stable solution. Gzara (2013) proposes a family of valid cuts and incorporates with an exact cutting plane algorithm for solving a bi-level network flow model. Bianco et al. (2015) study a novel toll setting policy and formulate a mathematical programming with equilibrium constraints where the government aims to minimize total risk and carriers intend to minimize travel cost. Taslimi et al. (2017) propose a bi-level network design model with the aim to minimize the maximum zone total risk and propose a greedy heuristic approach for large-size problems. Esfandeh et al. (2017) formulate the time-dependent network design problem based on altering carriers' departure times and route choices and extend the model that can consider consecutive time-based road closure policies and allow carriers to stop at the intermediate nodes.

2.2.3 Combined Facility Location and Network Design Problem in Non-hazmat Context

To the best of our knowledge, there are few papers related to combined facility location and network design problem in hazmat transportation; we review some relevant papers in non-hazmat context. The main difference between hazmat and non-hazmat problems is that hazmat problems usually need to be in the bi-level form with hierarchical decision-making.

Melkote and Daskin (2001b) investigate a generalized model that optimizes facility location and transportation network. Then they extend the model when facilities have a capacity constraint and present several classes of valid inequalities to strengthen its LP relaxation (Melkote and Daskin, 2001a). Ravi and Sinha (2006) propose an approximation algorithm for combined facility location and network design problem with minimizing facilities opening costs and transportation costs. Gelareh and Pisinger (2011) formulate a mixed integer linear programming for deep-sea liner service providers' locations and network design and propose a primal decomposition method. Contreras et al. (2012) present two mixed integer programming formulations which generalize the classical p-center problem in order to minimize the maximum customer-facility travel time. Ghaderi and Jabalameli (2013) present a model for the budget-constrained facility location–network design healthcare problem with minimizing multi-objectives of total travel costs and operating costs for facilities and network arcs. And a greedy heuristic is proposed based on simulated annealing and cutting plane method. Rahmaniani and Ghaderi (2013) propose a fix-and-optimize heuristic to solve bi-objective combined facility location and network design problem with capacitated arcs. Ghaderi (2015) studies a facility location-network design problem over several different time periods in order to minimize the maximum travel time between each pair of origin-destination and proposes an improved Variable Neighborhood Search.

2.2.4 Robust Optimization Approaches in Hazmat Transportation

In hazmat transportation problems, considering data uncertainty is necessary (Kwon et al., 2013). Stochastic programming methods are, however, less effective, because historical data are often insufficient to construct probability distributions for the risk exposure. When probability distributions of uncertain parameters are unknown, robust optimization is a useful technique (Bertsimas and Sim, 2003). Killmer et al. (2001) study a noxious facility location problem involving uncertainty by a robust optimization method. Sharma et al. (2009) formulate and solve the multi-objective robust network design problem with uncertain

demand. Berglund and Kwon (2014) consider a robust facility location and routing problem for hazardous materials management with the objective of minimizing the total cost and also analyze the impact of uncertainty in the demand and exposure risk. Xin et al. (2015) use robust optimization method to formulate a bi-level model under risk values uncertainty for designing hazmat transportation network. Sun et al. (2015) study a robust hazmat network design problem considering risk uncertainty and devise a heuristic method with Lagrangian relaxation. Sun et al. (2017) consider behavioral uncertainty from hazmat carriers and formulate a robust optimization problem, for which a cutting plane algorithm is devised.

2.3 The Robust Combined Facility Location-Network Design Problem

We consider a graph $G(\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of directed arcs. We assume that the sources of hazmat are at the known subset of nodes in the network, but the destinations (disposal facility) are not. We let $\mathcal{S}$ denote the set of hazmat shipments and $o(s)$ denote the origin node of shipment $s \in \mathcal{S}$. We want to determine the proper number and locations for constructing facilities from a set of candidate facility sites. Note that we assume disposal facilities do not generate hazmat; i.e., $\bigcup_{s\in\mathcal{S}} o(s) \cap \mathcal{M} = \emptyset$, where $\mathcal{M}$ denotes the set of candidate facility locations. At the same time, we will consider a road-ban policy by network designer. The upper level objective function is to minimize a linear combination of fixed facility cost and the worst-case exposure risk. The lower level objective function is to minimize the transportation cost for hazmat carriers. We assume that the hazmat carriers choose the least cost route to the nearest hazmat facility.

For shipment s, the expected number of trucks required is N^s. We let the anticipated risk induced by each truck for shipment s on arc (i, j) is R_{ij}^s. While the population exposure is a popular choice for the risk measure R_{ij}^s, one may use other metrics such as the accident probability and the environmental impact. We require the risk measure R_{ij}^s to hold linearity and additivity properties that ensures risk being measurable as the linear combination of metrics.

While one can estimate N^s and R_{ij}^s based on a survey and the national averages, the values of these two critical parameters are hardly known exactly (Berglund and Kwon, 2014). To address such data uncertainty, we employ a robust optimization approach. Following Berglund and Kwon (2014), we assume that the demand is given as an interval $[N^s, N^s + K^s]$ and the risk as $[R_{ij}^s, R_{ij}^s + Q_{ij}^s]$.

We denote the routing variable of hazmat carriers by $\mathbf{x}$, where $x_{ij}^s = 1$ if arc (i, j) is chosen for shipment s and $x_{ij}^s = 0$ otherwise. The worst-case total risk can be modeled as follows:

$$\max_{\mathbf{u}\in\mathcal{U},\mathbf{v}\in\mathcal{V}} \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} (N^s + K^s u^s)(R_{ij}^s + Q_{ij}^s v_{ij}) x_{ij}^s$$

where the uncertainty sets $\mathcal{U}$ and $\mathcal{V}$ are bounded. The uncertain variables $\mathbf{u}$ and $\mathbf{v}$ are constrained to stay within a specific range, and the total deviation from nominal values is limited by a budget of uncertainty. In particular, we define the uncertainty sets with the budget of uncertainty, Γ_u and Γ_v, as follows:

$$\mathcal{U} = \left\{ \mathbf{u} : \sum_{s\in S} u^s \le \Gamma_u, \quad 0 \le u^s \le 1 \right\}$$
$$\mathcal{V} = \left\{ \mathbf{v} : \sum_{(i,j)\in A} v_{ij} \le \Gamma_v, \quad 0 \le v_{ij} \le 1 \right\}.$$

Using the notation introduced in Table 2.1, we formulate the robust combined location-network design problem as the following bi-level optimization problem:

$$\underset{\mathbf{y},\mathbf{z}}{\text{minimize}} \qquad \left[w_1 \sum_{i\in\mathcal{M}} F_i y_i + w_2 \max_{\mathbf{u}\in\mathcal{U},\mathbf{v}\in\mathcal{V}} \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} (N^s + K^s u^s)(R_{ij}^s + Q_{ij}^s v_{ij}) x_{ij}^s \right] \tag{2.1}$$

$$\text{subject to} \qquad y_i \in \{0, 1\} \qquad \forall i \in \mathcal{M} \tag{2.2}$$

$$z_{ij} \in \{0, 1\} \qquad \forall (i, j) \in \mathcal{A} \tag{2.3}$$

where $\mathbf{x}$ solves

$$\underset{\mathbf{x}}{\text{minimize}} \qquad \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} c_{ij} x_{ij}^s \tag{2.4}$$

Table 2.1: Mathematical Notation

Sets	
$\mathcal{N}$	the set of nodes
$\mathcal{A}$	the set of arcs
$\mathcal{S}$	the set of hazmat shipments
$\mathcal{M}$	the set of candidate facility locations
$\mathcal{K}$	the set of chosen facility locations
Parameters	
c_{ij}	the cost of transportation though arc $(i,j) \in \mathcal{A}$
R_{ij}^s	the measure of exposure risk of shipment $s \in \mathcal{S}$ through arc $(i,j) \in \mathcal{A}$
$o(s)$	the node where hazmat are generated for shipment $s \in \mathcal{S}, o(s) \cap \mathcal{M} = \emptyset$
F_i	the cost of constructing a hazmat processing facility at node $i \in \mathcal{M}$
N^s	the number of trucks required for shipment $s \in \mathcal{S}$
Γ_u	the budget of uncertainty in the number of trucks
Γ_v	the budget of uncertainty in exposure risk
K^s	the width of the uncertainty in the number of trucks required by shipment $s \in \mathcal{S}$
Q_{ij}^s	the width of the uncertainty in the exposure risk through arc $(i,j) \in \mathcal{A}$
Variables	
x_{ij}^s	1, if arc $(i,j) \in \mathcal{A}$ is chosen for shipment $s \in \mathcal{S}$; 0, otherwise.
y_i	1, if a facility is located at node $i \in \mathcal{N}$; 0, otherwise.
z_{ij}	1, if arc $(i,j) \in \mathcal{A}$ is available for shipments; 0, otherwise.
u^s	the uncertainty variable for the number of trucks required for shipment $s \in \mathcal{S}$.
v_{ij}	the uncertainty variable for the exposure risk through arc $(i,j) \in \mathcal{A}$.

$$\text{subject to} \quad \sum_{j:(i,j)\in\mathcal{A}} x_{ij}^s - \sum_{j:(j,i)\in\mathcal{A}} x_{ji}^s \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -y_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S} \qquad (2.5)$$

$$x_{ij}^s \leq z_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \quad (2.6)$$

$$x_{ij}^s \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \quad (2.7)$$

Note that since facilities construction cost and exposure risk are not directly comparable, we will make a trade-off between these two parts of the objective function, i.e., set a dollar amount equal to a unit of exposure risk. If the decision maker is prone to avoid risk, he/she can set a higher dollar cost equal to a unit of risk, and vice versa. In the upper level objective

function (2.1), w_1 and w_2 represent the weights for cost and risk. The first part is the total facility construction cost and the second part represents the worst-case risk. Without loss of generality, we assume $(w_1, w_2) = (1, 1)$ for the rest of this chapter.

The lower level objective function (2.4) is to minimize the carriers' own shipment cost. Constraint (2.5) ensures that origin nodes must have net outflow of 1; when node i is selected as a facility ($y_i = 1$), node i can have net outflow of either -1 if node i is chosen as a destination or 0 otherwise; when node i is not selected as a facility ($y_i = 0$), node i is same as an intermediate node with zero net outflow; and all other intermediate nodes must have a zero balance. Constraint (2.6) means that selecting arc (i, j) is constrained by whether it is available ($z_{ij} = 1$) or not ($z_{ij} = 0$). Constraints (2.2), (2.3), and (2.7) represent that routing variable $\mathbf{x}$, location variable $\mathbf{y}$, and network design variable $\mathbf{z}$ are binary variables.

The bi-level optimization problem, where the upper-level problem is a min-max problem, can be formulated as a single-level optimization problem, shown in Section 2.5. The resulting single-level problem may be solved by off-the-shelf optimization solvers such as Gurobi and CPLEX, when the problem instance is small. For large problems, optimization solvers struggle with computational difficulty as shown in Section 2.6. There is also an issue with big-M in the single-level problem.

2.4 An Exact Solution Method

To solve the bi-level mixed integer program problem, we propose a cutting plane algorithm based on the cuts in Gzara (2013) and the idea of transforming location-network design problem into a pure network design problem from Melkote and Daskin (2001b). The nature of the cutting plane algorithm is to compare upper level objective (the Government's global goal) path and lower level objective (carrier's goal) path. When these two paths are same, an optimal solution is obtained. While the cutting plane algorithm can separate the lower-level problem as a subproblem from the upper-level master problem, the master problem is a computationally challenging problem, mainly due to the worst-case consideration in the

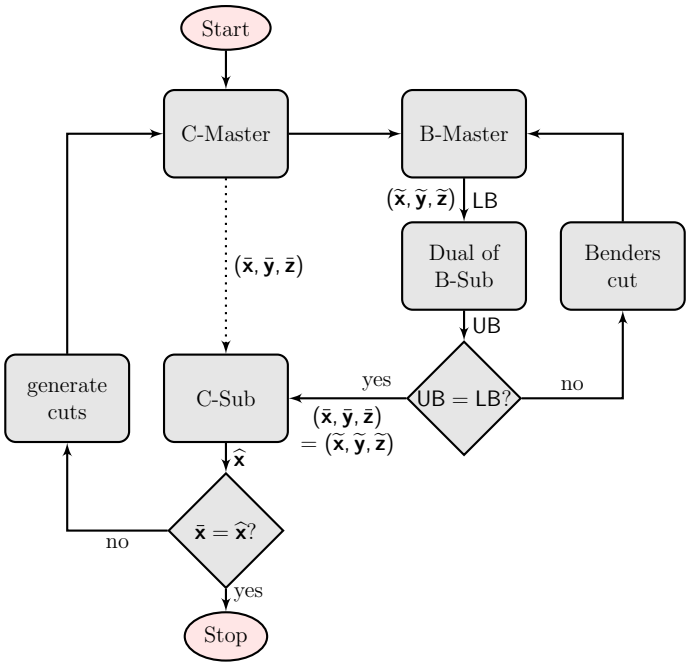


Figure 2.1: Flow Chart for the Cutting Plane Algorithm Combined with Benders Decomposition

upper-level objective. To tackle such difficulty, we use a Benders decomposition approach for solving the master problem. To distinguish master and subproblem from the cutting plane algorithm and Benders decomposition, we use C-Master/C-Sub and B-Master/B-Sub, respectively. We illustrate the entire computational framework in Figure 2.1. The dotted line represents the original flow in the cutting plane algorithm of Gzara (2013), which is replaced by Benders decomposition in this paper. Note that generated cuts in C-Master are carried over to B-Master, while Benders cuts are not carried over to C-Master.

2.4.1 Cutting Plane Algorithm

The master problem obtains the minimization of facility construction cost and the total shipment risk. The valid cuts (Section 2.4.2) will be added to C-Master iteratively. By adding cuts, network design variables z_{ij} can be changed to ensure carriers not to choose a certain arc. C-Master is firstly formulated as follows:

$$\underset{\mathbf{x},\mathbf{y},\mathbf{z}}{\text{minimize}} \left[\sum_{i \in \mathcal{M}} F_i y_i + \max_{\mathbf{u},\mathbf{v}} \sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} (N^s + K^s u^s)(R_{ij}^s + Q_{ij}^s v_{ij}) x_{ij}^s \right]$$

$$\text{subject to} \sum_{j:(i,j) \in \mathcal{A}} x_{ij}^s - \sum_{j:(j,i) \in \mathcal{A}} x_{ji}^s \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -y_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}$$

$$x_{ij}^s \leq z_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$x_{ij}^s \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$y_i \in \{0,1\} \qquad \forall i \in \mathcal{M}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}$$

$$\sum_{s \in S} u^s \leq \Gamma_u$$

$$\sum_{(i,j) \in A} v_{ij} \leq \Gamma_v$$

$$0 \leq u^s \leq 1 \qquad \forall s \in \mathcal{S}$$

$$0 \leq v_{ij} \leq 1 \qquad \forall (i,j) \in \mathcal{A}$$

additional cuts (2.23) (Section 2.4.2) added

Note that the above problem is a robust optimization problem for combined facility location-network design decisions, with additional cuts generated from the lower-level sub problem. To reformulate this problem as a single-level problem, we use dualization and linearization techniques introduced in Kwon et al. (2013). The inner maximization part can be expanded as follows:

$$\max_{\mathbf{u} \in \mathcal{U}, \mathbf{v} \in \mathcal{V}} \sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} (N^s + K^s u^s)(R_{ij}^s + Q_{ij}^s v_{ij}) x_{ij}^s$$

$$= \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} N^s R^s_{ij} x^s_{ij} + \max_{\mathbf{u}\in\mathcal{U},\mathbf{v}\in\mathcal{V}} \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} (N^s Q^s_{ij} v_{ij} + K^s R^s_{ij} u^s + K^s Q^s_{ij} u^s v_{ij}) x^s_{ij}$$

For any give $\mathbf{x}$, the inner maximization problem is equivalent as follows:

$$\begin{aligned}
\underset{\mathbf{u},\mathbf{v}}{\text{maximize}} \quad & \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} (N^s Q^s_{ij} v_{ij} + K^s R^s_{ij} u^s + K^s Q^s_{ij} u^s v_{ij}) x^s_{ij} \\
& \sum_{s\in\mathcal{S}} u^s \le \Gamma_u, \quad 0 \le u^s \le 1 \\
& \sum_{(i,j)\in\mathcal{A}} v_{ij} \le \Gamma_v, \quad 0 \le v_{ij} \le 1
\end{aligned}$$

By letting w^s_{ij} represent the quadratic term $u^s v_{ij}$ for each $(i,j) \in \mathcal{A}, s \in \mathcal{S}$, the above model can be linearized as follows:

$$\underset{\mathbf{u},\mathbf{v}}{\text{maximize}} \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} (N^s Q^s_{ij} v_{ij} + K^s R^s_{ij} u^s + K^s Q^s_{ij} w^s_{ij}) x^s_{ij}$$

$$\begin{aligned}
\text{subject to } & u^s \le 1 && \forall s \in \mathcal{S} && (\rho^s) \\
& v_{ij} \le 1 && \forall (i,j) \in \mathcal{A} && (\xi_{ij}) \\
& -u^s + w^s_{ij} \le 0 && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} && (\eta^s_{ij}) \\
& -v_{ij} + w^s_{ij} \le 0 && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} && (\pi^s_{ij}) \\
& \sum_{s\in\mathcal{S}} u^s \le \Gamma_u && && (\theta_u) \\
& \sum_{(i,j)\in\mathcal{A}} v_{ij} \le \Gamma_v && && (\theta_v) \\
& u^s \ge 0 && \forall s \in \mathcal{S} && \\
& v_{ij} \ge 0 && \forall (i,j) \in \mathcal{A} &&
\end{aligned}$$

The dual variables ρ^s, ξ_{ij}, η^s_{ij}, π^s_{ij}, θ_u and θ_v are introduced. The dual problem of the above problem becomes:

$$\underset{\boldsymbol{\rho},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\pi},\theta_u,\theta_v}{\text{minimize}} \quad \sum_{s\in S}\rho_s + \sum_{(i,j)\in\mathcal{A}}\xi_{ij} + \Gamma_u\theta_u + \Gamma_v\theta_v$$

$$\begin{aligned}
\text{subject to} \quad & \rho^s - \sum_{(i,j)\in\mathcal{A}}\eta^s_{ij} + \theta_u \geq \sum_{(i,j)\in\mathcal{A}} K^s R^s_{ij} x^s_{ij} && \forall s \in \mathcal{S} \\
& \xi_{ij} - \sum_{s\in\mathcal{S}}\pi^s_{ij} + \theta_v \geq \sum_{s\in\mathcal{S}} N^s Q^s_{ij} x^s_{ij} && \forall (i,j) \in \mathcal{A} \\
& \eta^s_{ij} + \pi^s_{ij} \geq K^s Q^s_{ij} x^s_{ij} && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \\
& \rho^s, \xi_{ij}, \eta^s_{ij}, \pi^s_{ij}, \theta_u, \theta_v \geq 0 && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}
\end{aligned}$$

We present the single-level linear optimization problem for C-Master:

$$\underset{\mathbf{x},\mathbf{y},\mathbf{z},\boldsymbol{\rho},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\pi},\theta_u,\theta_v}{\text{minimize}} \quad \sum_{i\in\mathcal{M}} F_i y_i + \sum_{(i,j)\in\mathcal{A}}\sum_{s\in\mathcal{S}} N^s R^s_{ij} x^s_{ij} + \sum_{s\in S}\rho_s + \sum_{(i,j)\in\mathcal{A}}\xi_{ij} + \Gamma_u\theta_u + \Gamma_v\theta_v$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{j:(i,j)\in\mathcal{A}} x^s_{ij} - \sum_{j:(j,i)\in\mathcal{A}} x^s_{ji} \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -y_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} && \forall i \in \mathcal{N}, s \in \mathcal{S} && (2.8) \\
& x^s_{ij} \leq z_{ij} && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} && (2.9) \\
& x^s_{ij} \in \{0,1\} && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} && (2.10) \\
& y_i \in \{0,1\} && \forall i \in \mathcal{M} && (2.11) \\
& z_{ij} \in \{0,1\} && \forall (i,j) \in \mathcal{A} && (2.12) \\
& \rho^s - \sum_{(i,j)\in\mathcal{A}}\eta^s_{ij} + \theta_u \geq \sum_{(i,j)\in\mathcal{A}} K^s R_{ij} x^s_{ij} && \forall s \in \mathcal{S} && (2.13) \\
& \xi_{ij} - \sum_{s\in\mathcal{S}}\pi^s_{ij} + \theta_v \geq \sum_{s\in\mathcal{S}} N^s Q^s_{ij} x^s_{ij} && \forall (i,j) \in \mathcal{A} && (2.14) \\
& \eta^s_{ij} + \pi^s_{ij} \geq K^s Q^s_{ij} x^s_{ij} && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} && (2.15)
\end{aligned}$$

$$\rho^s, \xi_{ij}, \eta^s_{ij}, \pi^s_{ij}, \theta_u, \theta_v \geq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \quad (2.16)$$

additional cuts (2.23) (Section 2.4.2) added

Let $\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{z}}$ be the solution of the C-master problem. The C-master problem is still difficult to solve; we use Benders decomposition to solve it. C-Master is divided into an integer Benders Master problem (B-Master) and a continuous Benders Sub problem (B-Sub). The decision variables are divided into two parts: binary variables x^s_{ij}, y_i, z_{ij} and continuous variables $\rho^s, \xi_{ij}, \eta^s_{ij}, \pi^s_{ij}, \theta_u, \theta_v$. The B-Sub problem generates a cut that is added to B-Master problem in every iteration. When the objectives of B-Master and B-Sub are same, the solutions $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$, and $\bar{\mathbf{z}}$ are obtained.

Fixing $\mathbf{y} = \bar{\mathbf{y}}$ and $\mathbf{z} = \bar{\mathbf{z}}$, we write the C-Sub problem as follows:

$$\underset{\mathbf{x}}{\text{minimize}} \quad \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} c_{ij} x^s_{ij}$$

$$\text{subject to} \quad \sum_{j:(i,j)\in\mathcal{A}} x^s_{ij} - \sum_{j:(j,i)\in\mathcal{A}} x^s_{ji} \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -\bar{y}_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}$$

$$x^s_{ij} \leq \bar{z}_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$x^s_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

Let $\widehat{\mathbf{x}}$ be the solution of the C-Sub problem, which represents the path which minimizes the transportation costs with the given facility location and network design.

2.4.2 Cut Generation

While Gzara (2013) has provided effective cut generation methods for the hazmat network design problem, our problem involves both network design and facility location variables. To apply the method of Gzara (2013) to our problem, we first transform the combined

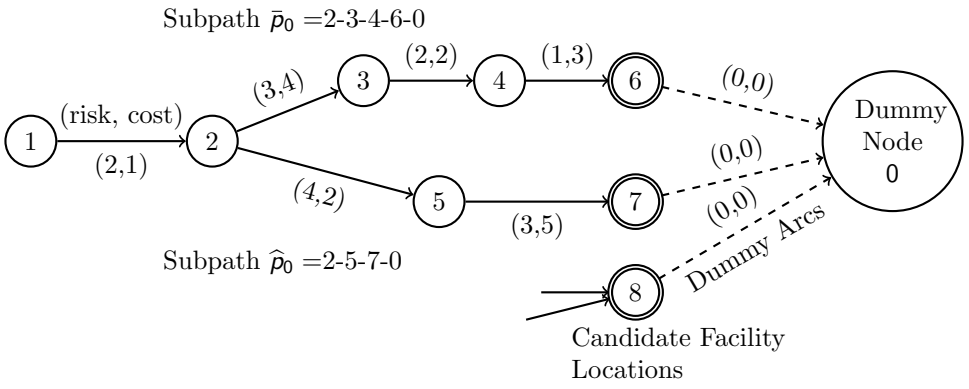


Figure 2.2: Conversion to a Pure Network Design Problem

facility location and network design problem to a pure network design problem (Melkote and Daskin, 2001b). As shown in Figure 2.2, all facility candidate locations are first connected to a dummy node via dummy arcs. Since the risk and transportation cost are zero in all dummy arcs, constructing a facility in a candidate location is equivalent to opening the corresponding dummy arc for traveling. By adding the dummy node, labeled as '0', and dummy arcs, labeled as $(k, 0)$ for each candidate location $k \in \mathcal{M}$, we obtain new sets of nodes and arcs as follows:

$$\mathcal{N}_0 = \mathcal{N} \cup \{0\}$$

$$\mathcal{A}_0 = \mathcal{A} \cup \{(k, 0) : k \in \mathcal{M}\}$$

As a result, we obtain an augmented graph $G_0(\mathcal{N}_0, \mathcal{A}_0)$, in which new "network design" variable z_{k0} for each $k \in \mathcal{M}$ corresponds to location variable y_k.

For each shipment s, two solutions $\bar{\mathbf{x}}$ and $\widehat{\mathbf{x}}$ utilize different paths. Among such two different paths, we obtain two distinct subpaths $\bar{p}$ and $\widehat{p}$ from $\bar{\mathbf{x}}$ and $\widehat{\mathbf{x}}$, respectively. Adding the dummy node to $\bar{p}$ and $\widehat{p}$, we obtain subpaths $\bar{p}_0$ and $\widehat{p}_0$ defined in $G_0(\mathcal{N}_0, \mathcal{A}_0)$, respectively. When the cuts suggested by Gzara (2013) are applied in $G_0(\mathcal{N}_0, \mathcal{A}_0)$, we obtain:

$$\sum_{(i,j)\in\bar{p}_0} x_{ij}^s \leq |\bar{p}_0| - 1 + u_{\text{new}} \tag{2.17}$$

$$u_{\text{new}} \leq x^s_{ij} \qquad \forall (i,j) \in \bar{p}_0 \tag{2.18}$$

$$\sum_{(i,j)\in\widehat{p}_0} z_{ij} \leq |\widehat{p}_0| - u_{\text{new}} \tag{2.19}$$

$$u_{\text{new}} \in \{0,1\} \tag{2.20}$$

where $|p|$ means the number of arcs in path p. We first show that the above cuts can be simplified.

Proposition 1. Inequalities in (2.17)–(2.20) hold if and only if

$$\sum_{(i,j)\in\widehat{p}_0} z_{ij} \leq |\widehat{p}_0| + |\bar{p}_0| - 1 - \sum_{(i,j)\in\bar{p}_0} x^s_{ij} \tag{2.21}$$

holds.

Proof. We consider each direction separately.

[$\Longrightarrow$] Summing inequalities (2.17) and (2.19), we obtain

$$\sum_{(i,j)\in\bar{p}_0} x^s_{ij} + \sum_{(i,j)\in\widehat{p}_0} z_{ij} \leq |\bar{p}_0| - 1 + |\widehat{p}_0|,$$

which is (2.21).

[$\Longleftarrow$] We now show that (2.21) implies (2.17)–(2.20). We consider two cases:

- When $\sum_{(i,j)\in\bar{p}_0} x^s_{ij} = |\bar{p}_0|$. Then $x^s_{ij} = 1$ for all $(i,j) \in \bar{p}_0$. Also (2.21) implies that

  $$\sum_{(i,j)\in\widehat{p}_0} z_{ij} \leq |\widehat{p}_0| - 1.$$

  For such $\mathbf{x}$ and $\mathbf{z}$, we can set $u_{\text{new}} = 1$ so that (2.17)–(2.20) hold.

- When $\sum_{(i,j)\in\bar{p}_0} x^s_{ij} \leq |\bar{p}_0| - 1$. Then $x^s_{ij} = 0$ for some $(i,j) \in \bar{p}_0$. Observe that the right-hand-side of (2.21) is greater than or equals to $|\widehat{p}_0|$. Since z_{ij} is binary, we have $\sum_{(i,j)\in\widehat{p}_0} z_{ij} \leq |\widehat{p}_0|$ by definition. Therefore by setting $u_{\text{new}} = 0$, we find that (2.17)–(2.20) hold.

□

Note that (2.21) can be written as

$$\sum_{(i,j)\in\widehat{p}_0}(1-z_{ij}) \geq 1-|\bar{p}_0|+\sum_{(i,j)\in\bar{p}_0} x_{ij}^s, \tag{2.22}$$

which has the following simple interpretation. If we want to flow $\mathbf{x}$ through subpath $\bar{p}_0$, i.e. $\sum_{(i,j)\in\bar{p}_0} x_{ij}^s = |\bar{p}_0|$, then at least one arc $(i,j) \in \widehat{p}_0$ must be closed or $\sum_{(i,j)\in\widehat{p}_0}(1-z_{ij}) \geq 1$.

Then we write the cut (2.22) in the original network $G(\mathcal{N},\mathcal{A})$.

Proposition 2. Let

$$\widehat{\delta}_k = \begin{cases} 1 & \text{if } \widehat{p} \text{ includes node } k \\ 0 & \text{otherwise} \end{cases}$$

for each $k \in \mathcal{M}$. Then the cut in (2.22) is equivalently written as

$$\sum_{(i,j)\in\widehat{p}}(1-z_{ij}) + \sum_{k\in\mathcal{M}} \widehat{\delta}_k(1-y_k) \geq 1-|\bar{p}|+\sum_{(i,j)\in\bar{p}} x_{ij}^s \tag{2.23}$$

for the original network $G(\mathcal{N},\mathcal{A})$.

Proof. If subpath $\bar{p}$ includes any facility location, we observe that

$$|\bar{p}_0| = |\bar{p}|+1$$
$$\sum_{(i,j)\in\bar{p}_0} x_{ij}^s = \sum_{(i,j)\in\bar{p}} x_{ij}^s + 1$$

since every shipment must flow to the dummy node. If subpath $\bar{p}$ does not involve a facility location, then

$$|\bar{p}_0| = |\bar{p}|$$

$$\sum_{(i,j)\in\bar{p}_0} x_{ij}^s = \sum_{(i,j)\in\bar{p}} x_{ij}^s.$$

Therefore, in both cases, we have

$$|\bar{p}_0| - \sum_{(i,j)\in\bar{p}_0} x_{ij}^s = |\bar{p}| - \sum_{(i,j)\in\bar{p}} x_{ij}^s.$$

With similar consideration, we also observe that

$$\sum_{(i,j)\in\widehat{p}_0} (1 - z_{ij}) = \sum_{(i,j)\in\widehat{p}} (1 - z_{ij}) + \sum_{k\in\mathcal{M}} \widehat{\delta}_k (1 - y_k).$$

Hence, we obtain a proof. □

2.4.3 Benders Decomposition for Solving C-Master

To solve the C-Master problem, we consider Benders Decomposition. The Benders Master (B-Master) problem contains binary variables x_{ij}^s, y_i, and z_{ij} and constraints that restrict the binary variables; namely, (2.8)–(2.12) and the cuts added in C-Master. We define the B-Master problem as follows:

$$\underset{\mathbf{x},\mathbf{y},\mathbf{z},d}{\text{minimize}} \quad \sum_{i\in\mathcal{M}} F_i y_i + \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} N^s R_{ij}^s x_{ij}^s + d$$

$$\text{subject to} \quad \sum_{j:(i,j)\in\mathcal{A}} x_{ij}^s - \sum_{j:(j,i)\in\mathcal{A}} x_{ji}^s \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -y_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}$$

$$x_{ij}^s \leq z_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$x_{ij}^s \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$y_i \in \{0,1\} \qquad \forall i \in \mathcal{M}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}$$

cuts (2.23) (Section 2.4.2) carried over from C-Master

additional Benders cuts (2.24) added

where d represents the remainder of the objective function that will be computed by sub-problems and constrained by Benders cuts. Let $\widetilde{\mathbf{x}}$, $\widetilde{\mathbf{y}}$, $\widetilde{\mathbf{z}}$, and $\widetilde{d}$ denote the optimal solutions of B-Master. Then B-Master gives a lower bound for C-Master. We let

$$\mathsf{LB} = \sum_{i \in \mathcal{M}} F_i \widetilde{y}_i + \sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} N^s R^s_{ij} \widetilde{x}^s_{ij} + \widetilde{d}.$$

The Benders Sub (B-Sub) problem contains continuous variables $\boldsymbol{\rho}, \boldsymbol{\xi}, \boldsymbol{\eta}, \boldsymbol{\pi}, \theta_u, \theta_v$ and constraints (2.13)–(2.16). The B-Sub problem is given by fixing $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$ with a solution of $\widetilde{\mathbf{x}}$, $\widetilde{\mathbf{y}}$, and $\widetilde{\mathbf{z}}$ by solving the B-Master problem. The B-Sub problem is defined as follows:

$$\underset{\boldsymbol{\rho},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\pi},\theta_u,\theta_v}{\text{minimize}} \quad \left[\sum_{s \in \mathcal{S}} \rho_s + \sum_{(i,j) \in \mathcal{A}} \xi_{ij} + \Gamma_u \theta_u + \Gamma_v \theta_v \right]$$

$$\begin{aligned}
\text{subject to} \quad & \rho^s - \sum_{(i,j) \in \mathcal{A}} \eta^s_{ij} + \theta_u \geq \sum_{(i,j) \in \mathcal{A}} K^s R_{ij} \widetilde{x}^s_{ij} && \forall s \in \mathcal{S} && (\alpha^s) \\
& \xi_{ij} - \sum_{s \in \mathcal{S}} \pi^s_{ij} + \theta_v \geq \sum_{s \in \mathcal{S}} N^s Q^s_{ij} \widetilde{x}^s_{ij} && \forall (i,j) \in \mathcal{A} && (\beta_{ij}) \\
& \eta^s_{ij} + \pi^s_{ij} \geq K^s Q^s_{ij} \widetilde{x}^s_{ij} && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} && (\gamma^s_{ij}) \\
& \rho^s, \xi_{ij}, \eta^s_{ij}, \pi^s_{ij}, \theta_u, \theta_v \geq 0 && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}
\end{aligned}$$

Because the optimality and valid cuts of B-Master problem can be defined by the dual variables of B-Sub problem, we formulate the dual for B-Sub. The dual variables α^s, β_{ij}, and γ^s_{ij} are introduced. The dual problem for B-Sub is presented as follows:

$$\underset{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}}{\text{maximize}} \quad \sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} (K^s R^s_{ij} \widetilde{x}^s_{ij} \alpha^s + N^s Q^s_{ij} \widetilde{x}^s_{ij} \beta_{ij} + K^s Q^s_{ij} \widetilde{x}^s_{ij} \gamma^s_{ij})$$

$$\begin{aligned}
\text{subject to} \quad & \alpha^s \leq 1 && \forall s \in \mathcal{S}
\end{aligned}$$

$$\beta_{ij} \leq 1 \qquad \forall (i,j) \in \mathcal{A}$$

$$-\alpha^s + \gamma_{ij}^s \leq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$-\beta_{ij} + \gamma_{ij}^s \leq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\sum_{s \in \mathcal{S}} \alpha^s \leq \Gamma_u$$

$$\sum_{(i,j) \in \mathcal{A}} \beta_{ij} \leq \Gamma_v$$

$$\alpha^s, \beta_{ij}, \gamma_{ij}^s \geq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

Let $\widetilde{\alpha}^s$, $\widetilde{\beta}_{ij}$, and $\widetilde{\gamma}_{ij}^s$ be the optimal solution of the Dual of the B-Sub problem. The following valid cut is added to the B-Master problem:

$$\sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} (K^s R_{ij}^s \widetilde{\alpha}^s + N^s Q_{ij}^s \widetilde{\beta}_{ij} + K^s Q_{ij}^s \widetilde{\gamma}_{ij}^s) x_{ij}^s \leq d. \tag{2.24}$$

We also obtain an upper bound for C-Master as follows:

$$\mathsf{UB} = \sum_{i \in \mathcal{M}} F_i \widetilde{y}_i + \sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} N^s R_{ij}^s \widetilde{x}_{ij}^s + \sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} (K^s R_{ij}^s \widetilde{x}_{ij}^s \widetilde{\alpha}^s + N^s Q_{ij}^s \widetilde{x}_{ij}^s \widetilde{\beta}_{ij} + K^s Q_{ij}^s \widetilde{x}_{ij}^s \widetilde{\gamma}_{ij}^s).$$

If $\mathsf{UB} = \mathsf{LB}$, then an optimal solution for C-Master is obtained.

2.5 A Single-Level Reformulation

We provide a single-level reformulation of the robust combined location-network design problem given in (2.1)–(2.7). We first replace the lower-level problem by its optimality conditions using techniques similar to the methods used by Arslan et al. (2018). Then we dualize and linearize the inner maximization problem for the worst-case consideration as done in Berglund and Kwon (2014). The resulting single-level reformulation involves a big-M like constant bounded by $\sum_{(i,j) \in \mathcal{A}} c_{ij}$. We will use this single-level reformulation as a benchmark for the cutting-plane method developed in Section 2.4.

2.5.1 Replacing the Lower-Level Problem by Optimality Conditions

Since the lower-level shortest path problem has the property of totally unimodular matrices (Kara and Verter, 2004), the binary variable x_{ij}^s can be relaxed to a nonnegative real number. We also introduce a dummy node 0 to transform the problem into a pure network design problem as done in Section 2.4.2. The lower-level problem can be written equivalently as follows:

$$\underset{\mathbf{x}}{\text{minimize}} \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} c_{ij} x_{ij}^s$$

$$\text{subject to} \sum_{j:(i,j)\in\mathcal{A}} x_{ij}^s - \sum_{j:(j,i)\in\mathcal{A}} x_{ji}^s \begin{cases} = 1 & \text{if } i = o(s) \\ +x_{i0} = 0 & \text{if } i \in \mathcal{M} \\ = -1 & \text{if } i = 0 \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S} \qquad (-\lambda_i^s)$$

$$(1 - z_{ij}) x_{ij}^s \leq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \qquad (-\mu_{ij}^s)$$

$$(1 - y_i) x_{i0}^s \leq 0 \qquad \forall i \in \mathcal{M}, s \in \mathcal{S} \qquad (-\mu_{i0}^s)$$

$$x_{ij}^s \geq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$x_{i0}^s \geq 0 \qquad \forall i \in \mathcal{M}, s \in \mathcal{S}$$

The dual variables $-\lambda_i^s$ and $-\mu_{ij}^s$ are introduced. The dual problem is:

$$\underset{\boldsymbol{\lambda},\boldsymbol{\mu}}{\text{maximize}} \sum_{s\in\mathcal{S}} \left(\lambda_0^s - \lambda_{o(s)}^s\right) \qquad (2.25)$$

$$\text{subject to} \quad -\lambda_i^s + \lambda_j^s - (1 - z_{ij})\mu_{ij}^s \leq c_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \quad (2.26)$$

$$-\lambda_i^s + \lambda_0^s - (1 - y_i)\mu_{i0}^s \leq 0 \qquad \forall i \in \mathcal{M}, \forall s \in \mathcal{S} \quad (2.27)$$

$$\mu_{ij}^s \geq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \quad (2.28)$$

$$\mu_{i0}^s \geq 0 \qquad \forall i \in \mathcal{M}, s \in \mathcal{S} \quad (2.29)$$

Using an approach similar to Arslan et al. (2018), we obtain the following result:

Proposition 3. Let $\bar{\mu} = \sum_{(i,j)\mathcal{A}} c_{ij}$. There exists an optimal solution for (2.25)–(2.29) with $\mu^s_{ij} = \mu^s_{i0} = \bar{\mu}$ for all $s \in \mathcal{S}$, $(i,j) \in \mathcal{A}$ and $i \in \mathcal{M}$.

Proof. By letting $\lambda^s_{o(s)} = 0$ without loss of generality, we obtain:

$$\underset{\boldsymbol{\lambda},\boldsymbol{\mu}}{\text{maximize}} \sum_{s\in\mathcal{S}} \lambda^s_0$$

$$\begin{aligned}
\text{subject to } & \lambda^s_j \le \lambda^s_i + c_{ij} + (1 - z_{ij})\mu^s_{ij} && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \\
& \lambda^s_0 \le \lambda^s_i + (1 - y_i)\mu^s_{i0} && \forall i \in \mathcal{M}, \forall s \in \mathcal{S} \\
& \mu^s_{ij} \ge 0 && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \\
& \mu^s_{i0} \ge 0 && \forall i \in \mathcal{M}, s \in \mathcal{S}
\end{aligned}$$

Note that $\boldsymbol{\mu}$ does not contribute to the objective function; therefore we can make $(1 - z_{ij})\mu^s_{ij}$ and $(1 - y_i)\mu^s_{i0}$ arbitrarily large to maximize λ^s_0. Since λ^s_0 represents a label for node d, we can bound μ^s_{ij} and μ^s_{i0} by $\bar{\mu}$. □

Therefore, the dual feasibility becomes:

$$\begin{aligned}
& \lambda^s_j \le \lambda^s_i + c_{ij} + (1 - z_{ij})\bar{\mu} && \forall (i,j) \in \mathcal{A}, s \in \mathcal{S} \\
& \lambda^s_0 \le \lambda^s_i + (1 - y_i)\bar{\mu} && \forall i \in \mathcal{M}, \forall s \in \mathcal{S}.
\end{aligned}$$

Note that $\bar{\mu}$ behaves like big-M constants. For the optimality condition, instead of the strong duality, we can use the reverse weak duality (Amaldi et al., 2011; Arslan et al., 2018) in the following form:

$$\sum_{s\in\mathcal{S}} \left(\lambda^s_0 - \lambda^s_{o(s)}\right) \ge \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} c_{ij} x^s_{ij}$$

Therefore, the robust combined location-network design problem becomes:

$$\underset{\mathbf{x},\mathbf{y},\mathbf{z},\boldsymbol{\lambda}}{\text{minimize}} \quad \left[\sum_{i\in\mathcal{M}} F_i y_i + \max_{\mathbf{u}\in\mathcal{U},\mathbf{v}\in\mathcal{V}} \sum_{(i,j)\in\mathcal{A}}\sum_{s\in\mathcal{S}} (N^s + K^s u^s)(R^s_{ij} + Q^s_{ij} v_{ij}) x^s_{ij}\right]$$

$$\text{subject to} \quad \sum_{j:(i,j)\in\mathcal{A}} x^s_{ij} - \sum_{j:(j,i)\in\mathcal{A}} x^s_{ji} \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -y_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}$$

$$x^s_{ij} \leq z_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\sum_{s\in\mathcal{S}} \left(\lambda^s_0 - \lambda^s_{o(s)}\right) \geq \sum_{(i,j)\in\mathcal{A}}\sum_{s\in\mathcal{S}} c_{ij} x^s_{ij}$$

$$\lambda^s_j \leq \lambda^s_i + c_{ij} + (1 - z_{ij})\bar{\mu} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\lambda^s_0 \leq \lambda^s_i + (1 - y_i)\bar{\mu} \qquad \forall i \in \mathcal{M}, \forall s \in \mathcal{S}$$

$$x^s_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$y_i \in \{0,1\} \qquad \forall i \in \mathcal{M}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}$$

$$\lambda^s_i \geq 0 \qquad \forall i \in \mathcal{N} \cup \{0\}.$$

2.5.2 Dualizing and Linearizing the Inner Maximization Problem

The inner maximization problem can be dualized and linearized as done in Section 2.4.1. Finally, we obtain the single-level reformulation of the robust combined location-network design problem as follows:

$$\underset{\mathbf{x},\mathbf{y},\mathbf{z},\boldsymbol{\lambda},\boldsymbol{\rho},\boldsymbol{\xi},\boldsymbol{\eta},\boldsymbol{\pi},\theta_u,\theta_v}{\text{minimize}} \left[\sum_{i\in\mathcal{M}} F_i y_i + \sum_{(i,j)\in\mathcal{A}}\sum_{s\in\mathcal{S}} N^s R^s_{ij} x^s_{ij} + \sum_{s\in\mathcal{S}} \rho^s + \sum_{(i,j)\in\mathcal{A}} \xi_{ij} + \Gamma_u\theta_u + \Gamma_v\theta_v\right]$$

$$\text{subject to} \quad \sum_{j:(i,j)\in\mathcal{A}} x_{ij}^s - \sum_{j:(j,i)\in\mathcal{A}} x_{ji}^s \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -y_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}$$

$$x_{ij}^s \leq z_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\sum_{s\in\mathcal{S}} \left(\lambda_0^s - \lambda_{o(s)}^s\right) \geq \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} c_{ij} x_{ij}^s$$

$$\lambda_j^s \leq \lambda_i^s + c_{ij} + (1 - z_{ij})\bar{\mu} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\lambda_0^s \leq \lambda_i^s + (1 - y_i)\bar{\mu} \qquad \forall i \in \mathcal{M}, \forall s \in \mathcal{S}$$

$$\rho^s - \sum_{(i,j)\in\mathcal{A}} \eta_{ij}^s + \theta_u \geq \sum_{(i,j)\in\mathcal{A}} K^s R_{ij}^s x_{ij}^s \qquad \forall s \in \mathcal{S}$$

$$\xi_{ij} - \sum_{s\in\mathcal{S}} \pi_{ij}^s + \theta_v \geq \sum_{s\in\mathcal{S}} N^s Q_{ij}^s x_{ij}^s \qquad \forall (i,j) \in \mathcal{A}$$

$$\eta_{ij}^s + \pi_{ij}^s \geq K^s Q_{ij}^s x_{ij}^s \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$x_{ij}^s \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$y_i \in \{0,1\} \qquad \forall i \in \mathcal{M}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}$$

$$\lambda_i^s \geq 0 \qquad \forall i \in \mathcal{N} \cup \{d\}$$

$$\rho^s, \xi_{ij}, \eta_{ij}^s, \pi_{ij}^s, \theta_u, \theta_v \geq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

The above problem is a mixed integer linear program (MILP) where all integer variables are binary. Off-the-shelf optimization solvers such as Gurobi and CPLEX can be used to solve small-size problem. As the size increases, however, the amount of time required by solvers grows rapidly.

2.6 Numerical Experiments

The experiments are done on the computer which runs 64-bit Windows 10 with 2.60GHz Intel Core (TM i5-7300U) CPU and 8 GB RAM. The cutting plane algorithm is coded in Julia 0.6.4 (Bezanson et al., 2012) and JuMP.jl optimization modeling package (Dunning et al., 2017) is used. The single-level reformulation in Section 2.5 is solved by calling Gurobi 7.5.2 solver with default setting.

Numerical analysis is performed on a set of data from Ravenna city in Italy (Erkut and Alp, 2007). The road network in Ravenna consists of 111 nodes and 143 arcs. The risk R_{ij}^s on arc (i, j) is calculated as the summation of exposure risk from all four types of hazmat (methanol, chlorine, gasoline, and LPG). The transportation cost c_{ij} for arc (i, j) is measured as the actual distance in meters. The demand for each shipment s is measured as truckloads, i.e., the number of trucks N^s.

The experiments are done on two sets of instances, small size and large size. In the small-size problems, there are 9 origins and the set size of candidate facility locations $\mathcal{M}$ is 5 and 10. In the large-size problems, there are 20 origins of hazmat shipment. We randomly choose 5, 10 and 15 as the set of candidate facility locations $\mathcal{M}$.

The comparison between objectives and running time of the cutting plane algorithm and Gurobi for the single-level reformulation are calculated as follows:

$$\%\text{Obj} = \frac{\text{Objective of Gurobi} - \text{Objective of cutting plane}}{\text{Objective of cutting plane}} \times 100 \tag{2.30}$$

$$\%\text{Time} = \frac{\text{Running time of Gurobi} - \text{Running time of cutting plane}}{\text{Running time of cutting plane}} \times 100 \tag{2.31}$$

2.6.1 Analysis on the Small-Size Instances

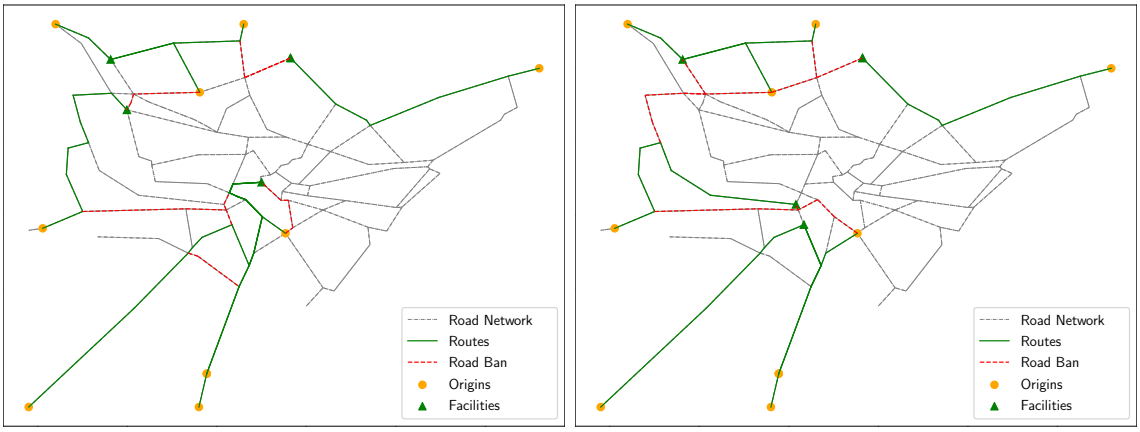
The objectives and running time on the small-size instances are shown in Table 2.2. For instances 3, 4, and 9, Gurobi fails to obtain a proven optimal solution in 3600s. The gaps between the incumbent solution and the best bound are 1.27%, 2.13%, and 0.71%,

respectively. The cutting plane algorithm can take less running time to get the proven optimal solutions except instances 5, 10, 15, and 20. Γ_u and Γ_v are much larger than those in other instances. As a result, the worst-case in the inner maximization problem happens when almost all u and v variables are set to 1. This makes the problem easier to solve for larger Γ_u and Γ_v values. So, for these instances, the optimal solution can be obtained in less running time by Gurobi than the cutting plane algorithm. The cutting plane algorithm performs better than Gurobi on 80% small-size instances in terms of running time. The average %Obj is 0.00% and the average %Time is 476.37%. The cutting plane algorithm and Gurobi can get optimal solution values for all small-size instances.

Table 2.2: Comparison Between the Solutions by the Cutting Plane Algorithm and Gurobi for the Single-Level Reformulation on Small-Size Ravenna Instances

Instance				Cutting-Plane		Single-Level		Comparison	
No.	$\lvert\mathcal{M}\rvert$	(K^s, Q^s_{ij})	(Γ_u, Γ_v)	Objective	Time(s)	Objective	Time(s)	%Obj	%Time
1	5	(N^s, R^s_{ij})	$(1, 1)$	19390	1.2	19390	2.9	0.00	141.7
2			$(3, 5)$	28966	43.0	28966	495.8	0.00	1053.0
3			$(5, 5)$	31222	66.5	31222[a]	3600.0	0.00	5313.5
4			$(5, 10)$	36221	1764.5	36221[a]	3600.0	0.00	104.0
5			$(10, 20)$	41814	1559.8	41814	655.4	0.00	-58.0
6		$(0.5N^s, 0.5R^s_{ij})$	$(1, 1)$	15628	1.1	15628	1.7	0.00	54.5
7			$(3, 5)$	19783	7.5	19783	36.8	0.00	390.7
8			$(5, 5)$	20779	11.3	20779	94.1	0.00	732.7
9			$(5, 10)$	22754	221.1	22754[a]	3600.0	0.00	1528.2
10			$(10, 20)$	24893	242.5	24893	96.3	0.00	-60.3
11	10	(N^s, R^s_{ij})	$(1, 1)$	11783	1.4	11783	1.9	0.00	35.7
12			$(3, 5)$	19491	2.6	19491	4.7	0.00	80.8
13			$(5, 5)$	20377	3.4	20377	5.3	0.00	55.9
14			$(5, 10)$	22678	5.0	22678	5.3	0.00	6.0
15			$(10, 20)$	25470	29.4	25470	6.0	0.00	-79.6
16		$(0.5N^s, 0.5R^s_{ij})$	$(1, 1)$	9424	1.3	9424	1.4	0.00	7.7
17			$(3, 5)$	12770	1.3	12770	1.6	0.00	23.1
18			$(5, 5)$	13158	1.4	13158	5.4	0.00	285.7
19			$(5, 10)$	13953	1.4	13953	1.4	0.00	0.0
20			$(10, 20)$	15251	20.1	15251	2.4	0.00	-88.1
Average								**0.00**	**476.37**

[a] The algorithm stopped in 3600s and the solution is not proven optimal.



(a) Instance 1 (b) Instance 11

Figure 2.3: Results on Small-Size Ravenna Instances

The results for Instance 1 and 11 are illustrated in Figure 2.3. The circles denote origins where hazmat is generated. The triangles represent chosen facilities sites. The green lines denote the routes which truck drivers choose. The red lines denote the roads which are not available for hazmat transportation.

2.6.2 Analysis on the Large-Size Instances

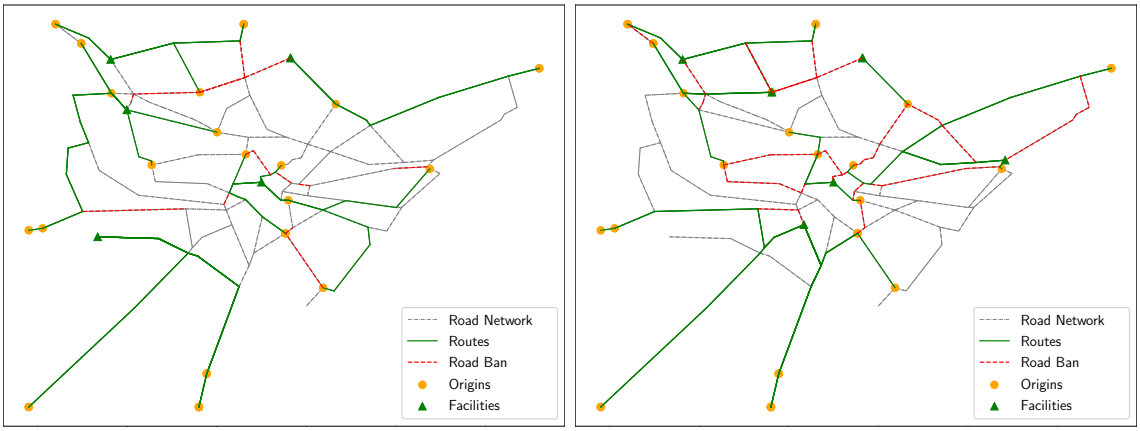
The objectives and running time on the large-size instances are shown in Table 3.3. The cutting plane algorithm and Gurobi can obtain optimal solutions for 100% and 58.33% large-size instances in 10800s, respectively. For instances 2 and 11, Gurobi obtains the incumbent solution that is equal to the value of the optimal solution. But the optimality of the incumbent solution can't be proven. The gaps between the incumbent solution and the best bound are 0.16% and 0.66%, respectively. For all instances, the cutting plane algorithm can take much less running time to get the proven optimal solutions. The average %Obj is 5.00% and the average %Time is 820.32%. The cutting plane algorithm outperforms the single-level reformulation solved by Gurobi in solution quality and running time. The results for Instance 1, 13, and 25 are illustrated in Figure 2.4.

The performance profile (Dolan and Moré, 2002) is used to compare different algorithms on the running times. The running time performance file for different algorithms is created

Table 2.3: Comparison Between the Solutions by the Cutting Plane Algorithm and Gurobi for the Single-Level Reformulation on Large-Size Ravenna Instances

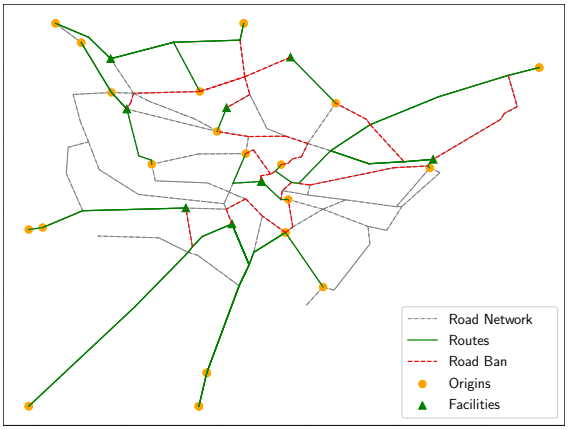
Instance				Cutting-Plane		Single-Level		Comparison	
No.	$\|\mathcal{M}\|$	(K^s, Q^s_{ij})	(Γ_u, Γ_v)	Objective	Time(s)	Objective	Time(s)	%Obj	%Time
1	5	(N^s, R^s_{ij})	$(1,1)$	90089	8.4	90089	155.4	0.00	1750.00
2			$(3,5)$	137010	279.8	137010[a]	10800.0	0.00	3759.90
3			$(5,5)$	143015	41.9	143015	389.0	0.00	828.40
4			$(5,10)$	168512	148.4	168512	717.3	0.00	383.36
5			$(10,20)$	200155	4832.4	200296[a]	10800.0	0.10	123.49
6			$(20,20)$	266569	652.0	266569	893.6	0.00	37.06
7		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	74187	12.5	74187	38.0	0.00	204.00
8			$(3,5)$	93607	22.6	93607	2695.3	0.00	11826.11
9			$(5,5)$	96609	13.8	96609	194.8	0.00	1311.59
10			$(5,10)$	106437	46.3	106437	122.1	0.00	163.71
11			$(10,20)$	119942	1233.6	119942[a]	10800.0	0.00	775.49
12			$(20,20)$	124410	994.2	124660[a]	10800.0	0.20	986.30
13	10	(N^s, R^s_{ij})	$(1,1)$	50908	993.0	59432[a]	10800.0	16.74	987.61
14			$(3,5)$	67665	1723.9	85216[a]	10800.0	25.94	526.49
15			$(5,5)$	71346	1878.0	89698[a]	10800.0	25.72	475.08
16			$(5,10)$	80241	1111.5	97450[a]	10800.0	21.45	871.66
17			$(10,20)$	95987	3621.7	98472[a]	10800.0	2.59	198.20
18			$(20,20)$	104626	1804.6	104626	2485.5	0.00	37.73
19		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	39968	1203.2	46563[a]	10800.0	16.50	797.61
20			$(3,5)$	46879	1704.6	56822[a]	10800.0	21.21	533.58
21			$(5,5)$	48655	2464.6	58829[a]	10800.0	20.91	338.20
22			$(5,10)$	52257	998.5	62170[a]	10800.0	18.97	981.62
23			$(10,20)$	58732	3709.4	63444[a]	10800.0	8.02	191.15
24			$(20,20)$	72117	1406.2	73233[a]	10800.0	1.55	348.84
25	15	(N^s, R^s_{ij})	$(1,1)$	48717	3.4	48717	16.8	0.00	394.12
26			$(3,5)$	65921	66.2	65921	100.8	0.00	52.27
27			$(5,5)$	70251	128.2	70251	223.7	0.00	74.49
28			$(5,10)$	77221	14.1	77221	19.8	0.00	40.43
29			$(10,20)$	92991	136.9	92991	175.2	0.00	27.98
30			$(20,20)$	100673	129.1	100673	225.4	0.00	74.59
31		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	38090	45.1	38090	66.7	0.00	47.89
32			$(3,5)$	45218	225.6	45218	713.0	0.00	216.05
33			$(5,5)$	47211	166.5	47211	204.5	0.00	22.82
34			$(5,10)$	50117	78.1	50117	134.7	0.00	72.47
35			$(10,20)$	56673	31.6	56673	42.4	0.00	34.18
36			$(20,20)$	59772	26.4	59772	36.2	0.00	37.12
Average								**5.00**	**820.32**

[a] The algorithm stopped in 10800s and the solution is not proven optimal.



(a) Instance 1

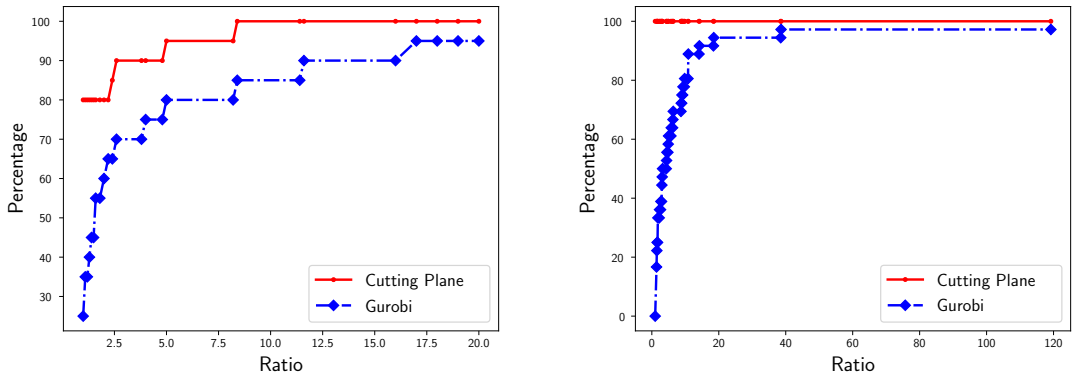
(b) Instance 13



(c) Instance 25

Figure 2.4: Results on Large-Size Ravenna Instances

by calculating the ratios of the running time of each algorithm and the minimum running time of all algorithms. The horizontal axis shows the ratios. The vertical axis shows the percentage of instances with a ratio that is less than or equal to the ratio on the horizontal axis. This indicates that the method has better performance when its profile is drawn in the upper left-hand graph. The running time performance profiles of the proposed cutting plane algorithm and Gurobi for the single-level reformation on small-size and large-size Ravenna instances are shown in Figure 2.5. Figure 2.5 indicates that the cutting plane algorithm has better running time performance than the single-level reformulation solved by Gurobi on small-size and large-size Ravenna Instances. In Figure 2.5(b), the profile of large-size instances is a straight line, that means the running times of the cutting plane algorithm are



(a) Small-Size Ravenna Instances (b) Large-Size Ravenna Instances

Figure 2.5: The Running Time Performance Profile of the Cutting Plane Algorithm and Gurobi for the Single-Level Reformulation

less than those by Gurobi for all instances. Compared large-size instances 5, 11, 17, and 23 with small-size instances 5, 10, 15, and 20, the difficulty caused by increasing problem size is more than the simplicity caused by large Γ_u and Γ_v values. Besides, the cutting plane algorithm takes less time in large-size instances 6, 12, 18, 24, 30, and 36, which have larger parameters values $(\Gamma_u, \Gamma_v) = (20, 20)$. Therefore, the advantage of the cutting plane algorithm becomes obvious as the problem size increases, especially in the running time.

2.6.3 Combined Model versus Sequential Model

We consider the *combined* robust model that optimizes facility locations and network design problem simultaneously. If we use techniques proposed in the previous papers in the literature review, the problem has to be solved in two phases. In Phase 1, robust facility locations problem is solved, and facility setup locations are determined. In Phase 2, robust network design problem is solved. We call this two-phase model the *sequential* model.

In this section, we use the single-level reformulation of bi-level robust facility location problem proposed by Kwon et al. (2013), shown in Appendix B.1. We obtain the optimal solution of facility location variables $\mathbf{y}^*$. When $y_i^* = 1$ for $i \in \mathcal{M}$, facility i is chosen to open. We let $\mathcal{K}$ be the set of chosen facility locations. Sun et al. (2015) considered a robust

Table 2.4: Comparison Between the Objectives for Combined and Sequential Model on Small-Size Ravenna Instances

Instance				Objective		
No.	$\|\mathcal{M}\|$	(K^s, Q^s_{ij})	(Γ_u, Γ_v)	Combined Model	Sequential Model	%Deviation
1	5	(N^s, R^s_{ij})	$(1,1)$	19390	20417	5.30
2			$(3,5)$	28966	30720	6.06
3			$(5,5)$	31222	33193	6.31
4			$(5,10)$	36221	36764	1.50
5			$(10,20)$	41814	42591	1.86
6		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	15628	15959	2.12
7			$(3,5)$	19783	20391	3.07
8			$(5,5)$	20779	21451	3.23
9			$(5,10)$	22754	22791	0.16
10			$(10,20)$	24893	25072	0.72
11	10	(N^s, R^s_{ij})	$(1,1)$	11783	12171	3.29
12			$(3,5)$	19491	20709	6.25
13			$(5,5)$	20377	21594	5.97
14			$(5,10)$	22678	23818	5.03
15			$(10,20)$	25470	25817	1.36
16		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	9424	9549	1.33
17			$(3,5)$	12855	12770	0.67
18			$(5,5)$	13158	13243	0.65
19			$(5,10)$	13953	13964	0.08
20			$(10,20)$	15251	15758	3.32
Average						**2.91**

network design problem only with the uncertainty in exposure risk. Based on their model, we revise a single-level form of robust network design model that considers uncertainty both in exposure risk and the number of shipment demand, shown in Appendix B.2. We obtain the optimal solution of routing variables $\mathbf{x}^*$ and network design variable $\mathbf{z}^*$.

To prove the benefits of the combined model, we compare it with the sequential model in the terms of objective function value $\sum_{i\in\mathcal{M}} F_i y_i^* + \sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} N^s R^s_{ij} x^{s*}_{ij} + \sum_{s\in\mathcal{S}} \rho^{s*} + \sum_{(i,j)\in\mathcal{A}} \xi^*_{ij} + \Gamma_u \theta^*_u + \Gamma_v \theta^*_v$. The comparison between objectives for combined model and sequential model is calculated as follows:

$$\%\text{Deviation} = \frac{\text{Objective of Sequential Model} - \text{Objective of Combined Model}}{\text{Objective of Combined Model}} \times 100 \tag{2.32}$$

The single-level reformulation models in Appendices B.1 and B.2 are solved by calling Gurobi 7.5.2 solver with default setting. Big-M is set as a constant bounded by $\sum_{(i,j)\in\mathcal{A}} c_{ij}$. The objectives of sequential model and %Deviation are shown in Tables 2.4 and 2.5. For small-size and large-size Ravenna city instances, the average %Deviation are 2.91 and 3.87, respectively. The objectives of the combined model are less than those of sequential model and the difference can be as large as 10.39%. In Table 2.5, we can observe that the difference is more obvious when $|\mathcal{M}|$ is smaller. When $|\mathcal{M}|$ is 5, 10, and 15, the average %Deviation are 6.52, 2.36, and 2.73, respectively.

This result indicates that when there are fewer choices of potential facility candidates, the value of combined decision-making becomes more significant. When there are more choices available for locations, there may exist a location that leads to a safe network even without network design policy. On the other hand, with fewer choices available for locations, such a favorable option may be unavailable; hence one needs to consider both location and network design decisions at the same time. When considering facility location and network design problem jointly, the leader can make a better decision with the aim of reducing the facility setup costs and hazmat exposure risk.

2.7 Concluding Remarks

In this chapter, a leader-follower decision problem is considered in the form of bi-level optimization. In the upper level, the leader aims to minimize the total facility construction costs and hazmat exposure risks by determining facility locations and available roads for hazmat transportation. The leader affects the followers who intend to minimize their trans-

Table 2.5: Comparison Between the Objectives for Combined and Sequential Model on Large-Size Ravenna Instances

Instance				Objective		
No.	$\lvert\mathcal{M}\rvert$	(K^s, Q^s_{ij})	(Γ_u, Γ_v)	Combined Model	Sequential Model	%Deviation
1	5	(N^s, R^s_{ij})	$(1,1)$	90089	99445	10.39
2			$(3,5)$	137010	148392	8.31
3			$(5,5)$	143015	157416	10.07
4			$(5,10)$	168512	176718	4.87
5			$(10,20)$	200155	211719	5.80
6			$(20,20)$	266569	273800	2.71
7		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	74187	79259	6.84
8			$(3,5)$	93607	100067	6.90
9			$(5,5)$	96609	103821	7.47
10			$(5,10)$	106437	111429	4.69
11			$(10,20)$	119942	125868	4.94
12			$(20,20)$	124410	130981	5.28
13	10	(N^s, R^s_{ij})	$(1,1)$	50908	51377	0.92
14			$(3,5)$	67665	68255	0.87
15			$(5,5)$	71346	72974	2.28
16			$(5,10)$	80241	80628	0.48
17			$(10,20)$	95987	100021	4.20
18			$(20,20)$	104626	109641	4.79
19		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	39968	40750	1.96
20			$(3,5)$	46879	47554	1.44
21			$(5,5)$	48655	49624	1.99
22			$(5,10)$	52257	52940	1.31
23			$(10,20)$	58732	61058	3.96
24			$(20,20)$	72117	75046	4.06
25	15	(N^s, R^s_{ij})	$(1,1)$	48717	50474	3.61
26			$(3,5)$	65921	66827	1.37
27			$(5,5)$	70251	71624	1.95
28			$(5,10)$	77221	78262	1.35
29			$(10,20)$	92991	95298	2.48
30			$(20,20)$	100673	103216	2.53
31		$(0.5N^s, 0.5R^s_{ij})$	$(1,1)$	38090	39847	4.61
32			$(3,5)$	45218	46124	2.00
33			$(5,5)$	47211	48098	1.88
34			$(5,10)$	50117	51048	1.86
35			$(10,20)$	56673	58208	2.71
36			$(20,20)$	59772	63602	6.41
Average						**3.87**

portation costs when designing the road network. We apply a robust optimization approach to deal with the uncertainty in the exposure risk and the demand. A bi-level integer programming model is formulated where the upper level is a min-max problem and the lower level is a shortest-path problem. We devise an exact algorithm that combines a cutting plane algorithm with Benders decomposition and derive a single-level reformulation. Comparisons between two approaches are made on the Ravenna city data, in terms of objectives and the running time. The analysis on small and large size instances demonstrates that the proposed cutting plane algorithm performs much better than Gurobi as the problem size increases. The proposed cutting plane algorithm is an effective exact method for solving the robust combined facility location-network design problem.

A couple of directions for future research are suggested. First, uncertainty on origin locations can be considered. In this chapter, we assume that all origin nodes are exactly known. Since the hazmat facility location problem is for long-term decision, considering new hazmat origins in the future will lead to an important problem. Second, hazmat trips to locations other than the hazmat facilities can be incorporated. Although we consider hazmat trips to hazmat facilities only in this chapter, there are also hazmat trips to other destinations. Hazmat network design policies will certainly impact not only trips to hazmat facilities, but also all other general hazmat trips. Therefore, incorporating both types of hazmat trips within a single modeling framework is a valuable research direction.

Chapter 3: An Adaptive Large Neighborhood Search Method for Rebalancing Free-Floating Electric Vehicle Sharing Systems

3.1 Introduction

Electric vehicle (EV) sharing systems are a promising solution for smarter and more sustainable urban mobility. Vehicle sharing systems have the potential to become a viable alternative to private car ownership, leading to more efficient utilization of vehicles and parking sites. Moreover, the use of EVs in vehicle sharing systems can offer an effective solution to curbing greenhouse gas emissions in urban transportation. Accordingly, many vehicle sharing companies, such as Zipcar and car2go, are developing rapidly with the traction of increasing vehicle rental demands. As of 2019, the global carsharing market exceeded USD 2.5 billion and is estimated to reach up to USD 9 billion in 2026 (Preeti and Prasenjit, 2019).

Unlike traditional carsharing systems that are based on fixed stations, free-floating carsharing systems allow users to pick up any available vehicle wherever and whenever they need it and return them at any permissible parking location within a designated service area. As of 2016, free-floating EV sharing systems (FFEVSS) are being operated in about 34 cities across nine countries (Kortum et al., 2016). An important feature of such vehicle sharing systems is that they allow users to make a customized one-way trip with the mobile apps, providing more flexible service (Wielinski et al., 2015). It is reported that this type of carsharing system reduces traffic congestion in urban areas (Becker et al., 2018; Le Vine and Polak, 2019). In North America, it is estimated that 11–19% of carsharing participants have sold their private vehicles after joining a carsharing program (Shaheen and Cohen, 2007). Moreover, carsharing improves the quality of mobility service compared to public transportation (Mattia et al., 2019).

Despite such clear benefits of FFEVSS, they also pose multiple operational challenges. First, the vehicles should be available at the right location at the right time based on the customers' demand patterns, as one-way trip demands can cause a severe spatial imbalance between vehicle availability and trip demands. Second, the low charging level of EVs may trigger customers' range anxiety and lower the service demand, and hence, the EVs should be frequently and sufficiently charged (Weikl and Bogenberger, 2015; Noel et al., 2019). Indeed, for a successful operation of FFEVSS, it is essential to develop an efficient EV relocation plan to resolve the imbalance and charging issues.

There are two key decisions in the FFEVSS rebalancing problem: (1) EV relocation and (2) shuttle routing. EV relocation decides which EV should be relocated from its origin (i.e., supply location) to which demand location, and also, whether and where it has to be charged. This operation can be particularly complex and costly, because EVs cannot be moved in a batch on one carrier, unlike other shared mobility systems such as bikes or scooters. In addition, an EV relocation operation requires a driver to drive the vehicle to its destination, and therefore, supporting vehicles, which we call shuttles, are employed to drop off and pick up the drivers. Hence, a shuttle routing problem arises. Specifically, the shuttle routing problem determines how such shuttles pick up and drop off drivers to satisfy the planned EV relocations. For an efficient operation, these two decisions should be made simultaneously rather than sequentially, which creates additional layers of computational challenge.

In this chapter, we develop an adaptive large neighborhood search (ALNS) algorithm to solve the FFEVSS rebalancing problem. ALNS, first proposed by Ropke and Pisinger (2006), is a well-known iterative metaheuristic framework that has been popularly applied to solving various vehicle routing problems. One of the key characteristics of ALNS is that it partially destroys an incumbent solution and repairs it to construct a better solution in each iteration. Normally, a small number of destroy and repair methods are used, and the choice of the method is determined adaptively as the algorithm proceeds. Applying ALNS to our problem setting is not at all trivial or straightforward. As two decisions of EV relocation

and shuttle routing are tangled in one problem, the destroy and repair methods should also be able to handle such complexity. We propose multiple destroy and repair methods that are adequate to the problem setting that we focus in this chapter. We then propose a way to modify our ALNS algorithm to demonstrate how it can accommodate various operational environments. For instance, we demonstrate how our ALNS algorithm can be adopted for the case where each driver carries a small personal mobility vehicle such as a scooter or Segway instead of being transported by a shuttle. In this case, our method routes the drivers directly, rather than routing the shuttles. We present how we may modify our main algorithm to create a variant.

One important strength of our ALNS algorithm, compared to other types of computational methods available for the FFEVSS rebalancing problems in the literature, is its flexibility to be applied in a dynamic environment. Because ALNS algorithm continuously improves the current route by destroying and repairing solutions, any change in the current system can be easily considered within the destroy and repair procedures. However, the ALNS framework applied to the EV relocation and shuttle routing problem should consider the hierarchy and interdependency of the two decisions to be made. Our key methodological contributions include developing efficient methods to handle such hierarchy and interdependency within the popular ALNS framework.

We also conduct numerical experiments for which we use randomly generated instances of several sizes as well as actual data from an FFEVSS operated in the city of Amsterdam. Haider et al. (2019) and Bogyrbayeva et al. (2021) proposed an exchange-based neighborhood-search method (EBNSM) and a Reinforcement Learning (RL) approach to solve the same problem, respectively. With EBNSM and RL being benchmark methods, we demonstrate the effectiveness and efficiency of our ALNS algorithm.

The remainder of the chapter is organized as follows. In Section 3.2, the related literature is reviewed. In Section 3.3, the problem statement and mathematical formulation are presented in detail. Two benchmark methods are described briefly in Section 3.4. The

computational method based on ALNS, our main methodological contribution, is presented in Section 3.5. The modification of ALNS in problem variants are presented in Section 3.6. In Section 3.7, the experimental results validate the performance of ALNS. We conclude this chapter in Section 3.8.

3.2 Literature Review

Since the inception of car2go operation, who started a free-floating carsharing system in Ulm, Germany in April 2009 (Firnkorn and Müller, 2011), many models and approaches have been proposed to optimize the operations in FFEVSS. In this section, we review the literature in this stream within the past decade.

Some papers focused only on vehicles relocation and did not take personnel allocation into account. Cepolina and Farina (2012) developed a random search algorithm to optimize the vehicles number and their distribution in the operating region with minimizing vehicle costs and waiting times. The method tried to narrow the difference between the number of available vehicles and users' demands in a time period. Weikl and Bogenberger (2013) implemented the online optimization module to collect and measure the current demand and vehicle distribution and predicted the future demand in regards to an offline module. Then, they proposed a mesoscopic relocation algorithm to relocate vehicles on two levels (macroscopic segment level and microscopic individual vehicle level) in the free-floating carsharing system. Herrmann et al. (2014) investigated different relocation strategies to balance cars distribution for the increasing customers' acceptance in the free-floating carsharing system and proposed a discrete-event simulation model that was evaluated by testing on car2go real case data. Jorge et al. (2014) developed a mathematical model to relocate vehicles to maximize the profit and a simulation model to discuss different real-time relocation policies. The results showed that the profit increased after operating vehicles relocations. Wielinski et al. (2015) proposed a practice-ready model for free-floating carsharing systems combined with the charging of electric cars and the refueling of conventional vehicles. They used two types

of relocation: intra-zone relocation, where vehicles move within them, and inter-zone relocation, where vehicles move between them. Boyacı et al. (2015) developed a multi-objective MILP model for one-way EV sharing systems that considered EV relocation and recharging requirements. A branch-and-bound approach was used to analyze the trade-off between operators' and customers' benefits.

Nevertheless, an inevitable practical problem in vehicle relocation is that each vehicle movement requires one worker who drives from the supply node to the demand location. Thus, EV relocation assignment requires a simultaneous personnel assignment. More recently, some researchers considered the integration of vehicles relocation and personnel assignment. Di Febbraro et al. (2012) used a discrete event systems model to represent carsharing system and proposed a user-based approach on an optimal relocation policy in a rolling horizon framework; the aiming was to maximize the operator's profit by the minimum number of required staff and relocate the least number of vehicles. Nourinejad and Roorda (2015) integrated two models of multi-traveling salesman problem for jointly optimizing vehicle relocation and personnel allocation. Nourinejad et al. (2015) further formulated two IP models for the hybrid system that has both one-way and two-way trips. The first model was built for tactical planning and calculated the number of required cars. The second model decided to accept user requests who profit the most for the service provider. For minimizing the total cost, Santos and Correia (2015) developed a MIP model for the real-time one-way carsharing system, which considered the maintenance, vehicle relocation, and personnel operation. Boyacı et al. (2017) further considered some hard constraints, such as parking station numbers and vehicle capacity limitation, and the service quality. A multi-objective MIP model was proposed and involved three sub mathematical models: station clustering, operations optimization, and personnel flow. The simulation framework was developed to balance the cost of vehicle relocation, workers relocation, and service level. In order to maximize the total profit, Bruglieri et al. (2017) proposed a Ruin-and-Recreate metaheuristic to solve the free-floating electric vehicle relocation problem where the workers use folding

bicycles to relocate vehicles. Zhao et al. (2018) formulated a MILP model to describe the EV rebalancing and workers relocation with spatial-time-dependent customers' reservations. A three-phase implementing algorithm was developed based on Lagrangian relaxation combined with dynamic programming and a greedy algorithm. Kypriadis et al. (2018, 2020) studied the minimum walking car repositioning problem where the workers walk to undertake relocation assignments. The charging of EVs and refueling of conventional cars were taken into account. The vehicle relocation approach aimed to minimize the relocation cost by minimizing the walking distance. Bruglieri et al. (2019) also proposed another metaheuristic, Adaptive Large Neighborhood Search metaheuristic, for the same problem. And ALNS method was proved to perform better than Tabu Search, Ruin and Recreate metaheuristic, and Mixed Integer Linear Programming. Haider et al. (2019) formulated a MIP model for the relocation operations in the sequential and synchronized approach and proposed the exchange-based neighborhood-search method (EBNSM) for large-scale problems. Bogyrbayeva et al. (2021) proposed a reinforcement learning approach for rebalancing EVs by considering charging in the free-floating electric vehicle sharing systems (FFEVSS). They focused on the shuttle routing problem and formulated the shuttles routes using a multi-agent reinforcement learning framework. EVs are relocated to the nearest available charger or demand node.

As stated above, a variety of methods have been proposed for solving the rebalancing free-floating carsharing system. To the best of our knowledge, only a few papers discussed EV relocation and personnel allocation jointly for the free-floating EV sharing system with the consideration of charging EVs, shuttles routing of transporting workers.

3.3 Problem Statement

We introduce our problem formally with the notation listed in Table 3.1. The service area is constrained in a region with $|\mathcal{N}|$ nodes. One node means a parking lot for an EV. The supply nodes are the places where the excess available EVs park and no customers will

Table 3.1: Mathematical Notation

Sets	
$\mathcal{N}$	Set of all original nodes, $\mathcal{N} = \mathcal{S} \cup \mathcal{D} \cup \mathcal{C}$
$\mathcal{N}'$	Set of all nodes including depot start node {0}, end node {N+1} and dummy charge nodes, $\mathcal{N}' = \mathcal{N} \cup \{0\} \cup \{N+1\} \cup \mathcal{C} \cup \mathcal{C}^+$
$\mathcal{S}$	Set of supply nodes, $\mathcal{S} = \mathcal{S}^n \cup \mathcal{S}^c$
$\mathcal{S}^n$	Set of supply nodes where EVs do not require charging
$\mathcal{S}^c$	Set of supply nodes where EVs require charging
$\mathcal{D}$	Set of demand nodes
$\mathcal{C}$	Set of real charge nodes
$\mathcal{C}'$	Set of real and paired dummy charge nodes
$\mathcal{C}^+$	Set of dummy charge process nodes
$\mathcal{K}$	Set of shuttles
$\mathcal{P}$	Set of all feasible EV relocation paths
$\phi(i)$	Set of paths that contain node i, $\{p \in \mathcal{P} : i \in p\}$
Parameters	
v_e	Average EV speed
v_s	Average shuttle speed
l_s	Initial battery percentage of EV at supply s
β	Charging time for one battery percentage
K	Number of shuttles $K = \lvert\mathcal{K}\rvert$
P	Number of all workers
W	Capacity of workers onboard one shuttle (exclude the workers who drive shuttles)
d_{ij}	Distance of arc (i, j)
M	A sufficient large positive number
H	A sufficient large positive number for creating dummy charge nodes
Variables	
x_p	1, EV is relocated along path $p \in \mathcal{P}$; Otherwise, 0.
y_{ij}	1, if a shuttle comes through arc(i, j) as part of route; Otherwise, 0.
z_i	The number of workers onboard a shuttle after this shuttle leaves node i.
τ_i	The time when a shuttle arrives node i
e_i	The time when an EV arrives node i

pick up EVs at those nodes on the next day. Based on the booking orders, the customers will pick up EVs at the demand nodes. Now there are no available EVs at these nodes. Thus, the sharing system company is required to relocate EVs from supply nodes to demand nodes to satisfy the customers' needs.

In this problem, two decisions are made: EV relocation and shuttles routing. EV relocation is divided into two cases: (1) EV can move directly from a supply node $s \in \mathcal{S}^n$ to a demand node when it has sufficient battery energy (more than the minimum battery level); and (2) When an EV does not have enough battery energy (less than minimum battery level). EV at supply node $s \in \mathcal{S}^c$ is required to go to a charge station $c \in \mathcal{C}$ to be charged fully before going to the demand node $d \in \mathcal{D}$. Shuttles routing determines the sequence of visiting nodes. Shuttles transport workers to supply nodes and charge stations. The workers will be picked up at the demand nodes and charge stations. Since the charging process takes a long time, it is assumed that the worker has two choices: 1) To be picked up by one shuttle to do other EV relocation assignments. Another worker will come to this charge station and drive EV to the demand node. 2) To stay at the charge station and wait for charging. Then the worker drives this EV to the demand node. The number of shuttles is K, and the capacity of workers onboard one shuttle is W. All shuttles leave the depot with a full load and return to the depot with all workers. The objective function is to minimize the total time spent in the system, i.e., makespan. The makespan is calculated as the time between shuttles leaving and all returning to the depot.

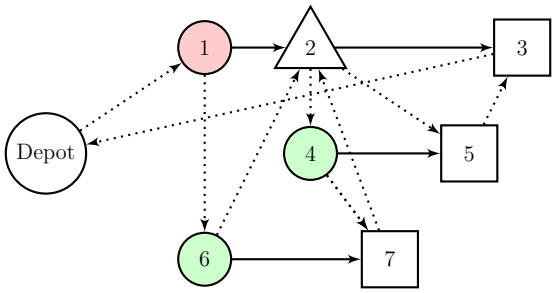


Figure 3.1: FFEVSS Example. (Circles, triangles, and squares represent supply nodes, charge stations, and demand nodes, respectively. Red circle means that EV requires charging; green circle means EV does not require charging; The solid lines represent EV relocation; the dotted lines represent a single shuttle route)

For example, in Figure 3.1, there is one shuttle and two workers. EV relocation contains 1-2-3, 4-5, and 6-7. The shuttle route is depot-1-6-2-4-7-2-5-3-depot. The shuttle starts from the depot and drops off 1st worker at supply 1. Then it goes to supply 6 and drops off

2nd worker. It picks up 1st worker at charge station 2 and transports him to supply 4. It continues to pick up 2nd worker at demand 7. The shuttle returns to charge station 2 and dispatches 2nd worker to drive the EV from node 2 to demand 3. Then, it picks up 1st and 2nd workers at demand 5 and 3. Finally, the shuttle carries all workers back to the depot.

3.3.1 Mathematical Model for EV Relocation

The dummy charge node-set $\mathcal{C}'$ is introduced to deal with multiple visits to each charge station. Because charge stations can serve many EVs and can be visited many times, each charge station needs sufficient enough dummy nodes. These dummy nodes have the same location coordinates as the real charge station. Define H as the number of dummy charge nodes for each real charge node. The depot end node is labeled as $N+1$, so the dummy charge nodes are labeled starting from $N+2$. For example, the dummy charge node for the first charge node is $\{N+2, N+3, .., N+1+H\}$. Let the total charge nodes set be $\mathcal{C}' = \mathcal{C} \cup \{N+2, N+3, ..., N+1+|\mathcal{C}| \times H\}$.

The paired dummy charge process node-set $\mathcal{C}^+ = \{c^+ : c \in \mathcal{C}'\}$ is introduced to deal with the charging process, where c^+ represents a copy of c. The charge time for EV at s is calculated from the initial battery percentage $I_s\%$ to 100% and is assumed to equal to $\beta \times (100 - I_s)$. Because the charging takes a long time, the workers can be dispatched to do other tasks. The charging process is described in Figure 3.2, EV goes to charge station c for charging, and after finishing charge, it goes to a dummy charge process node c^+. The pairs of charger-dummy charge process nodes $c \in \mathcal{C}', c^+ \in \mathcal{C}^+$ are designated to receiving workers and dispatching shuttles.

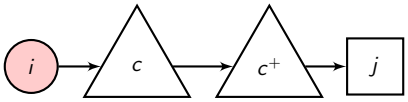


Figure 3.2: Relocation Decision. In a relocation with charge model, one relocation operation involves a stopover at a pair of charger-dummy charge process nodes

We can enumerate all the possible EV relocation paths. If an EV does not require charging, the path is defined as directly from any supply node to any demand node $p = \{s \to d : s \in \mathcal{S}^n, d \in \mathcal{D}\}$ and the number of paths is $|\mathcal{S}| \times |\mathcal{D}|$. If an EV requires charging, the path is defined as $p = \{s \to c \to d : s \in \mathcal{S}^c, c \in \mathcal{C}', d \in \mathcal{D}\}$ and the number of paths is $|\mathcal{S}| \times |\mathcal{C}'| \times |\mathcal{D}|$. The set of all possible EV relocation paths $\mathcal{P}$ is defined as

$$\mathcal{P} = \{s \to d : s \in \mathcal{S}^n, d \in \mathcal{D}\} \cup \{s \to c \to d : s \in \mathcal{S}^c, c \in \mathcal{C}', d \in \mathcal{D}\}$$

The mathematical formulation of EV relocation is described as follows:

$$\sum_{p \in \phi(i)} x_p = 1 \qquad \forall i \in \mathcal{D} \tag{3.1}$$

$$\sum_{p \in \phi(i)} x_p \leq 1 \qquad \forall i \in \mathcal{S} \cup \mathcal{C}' \tag{3.2}$$

$$x_p \in \{0, 1\} \qquad \forall p \in \mathcal{P} \tag{3.3}$$

$$e_d \geq e_s + \frac{d_{sd}}{v_e} \qquad \forall p = \{s \to d\} \in \mathcal{P} : s \in \mathcal{S}^n \tag{3.4}$$

$$e_c \geq e_s + \frac{d_{sc}}{v_e} \qquad \forall p = \{s \to c \to d\} \in \mathcal{P} : s \in \mathcal{S}^c \tag{3.5}$$

$$e_{c^+} \geq e_c + \beta(100 - l_s) \qquad \forall p = \{s \to c \to d\} \in \mathcal{P} : c \in \mathcal{C}' \tag{3.6}$$

$$e_d \geq e_{c^+} + \frac{d_{cd}}{v_e} \qquad \forall p = \{s \to c \to d\} \in \mathcal{P} \tag{3.7}$$

$$e_i \geq 0 \qquad \forall i \in \mathcal{N}' \tag{3.8}$$

Each demand node is visited exactly once, so the inflow of each demand node $i \in \mathcal{D}$ is equal to 1 (Constraint (3.1)). Each supply node and charge station is visited less than or equal to 1 (Constraint (3.2)). If EV is assigned from s to d, the time when EV arrives node d is more than or equal to EV's arrival time at node s plus the movement time (Constraint (3.4)). If EV moves along the path $s \to c \to d$, EV's arrival time at node c is the arrival time at node s plus movement time from s to c (Constraint (3.5)); EV arrives at dummy

charge node c^+ after adding a charging time $\beta(100 - l_s)$ (Constraint (3.6)); EV's arrival time at demand d is arrival time at dummy charge node c^+ plus movement time from c to d (Constraint (3.7)). Constraints (3.3) and (3.8) restrict that EV relocation path decision variable x^p is binary and EV arrival times e_i are nonnegative continuous variables.

3.3.2 Mathematical Model for Shuttle Routing

The shuttles routing problem is described as follows:

$$\sum_{(0,j)\in\mathcal{A}} y_{0j} = \sum_{(i,N+1)\in\mathcal{A}} y_{i,N+1} = K \tag{3.9}$$

$$\sum_{(i,j)\in\mathcal{A}} y_{ij} - \sum_{(j,i)\in\mathcal{A}} y_{ji} = 0 \qquad \forall j \in \mathcal{N}' \setminus \{0, N+1\} \tag{3.10}$$

$$n_j \geq n_i + |\mathcal{N}'| y_{ij} - (|\mathcal{N}'| - 1) \qquad \forall (i,j) \in \mathcal{A} \tag{3.11}$$

$$\sum_{(i,j)\in\mathcal{A}} y_{ij} \leq 1 \qquad \forall j \in \mathcal{S} \cup \mathcal{C}' \tag{3.12}$$

$$\sum_{(i,j)\in\mathcal{A}} y_{ij} = 1 \qquad \forall j \in \mathcal{D} \tag{3.13}$$

$$\sum_{(i,c)\in\mathcal{A}} y_{ic} = \sum_{(i,c^+)\in\mathcal{A}} y_{ic^+} \qquad \forall c \in \mathcal{C}' \tag{3.14}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A} \tag{3.15}$$

$$\tau_j \geq \tau_i + \frac{d_{ij}}{v_s} - M(1 - y_{ij}) \qquad \forall (i,j) \in \mathcal{A} \tag{3.16}$$

$$\tau_i \geq 0 \qquad \forall i \in \mathcal{N}' \tag{3.17}$$

$$z_0 = W \tag{3.18}$$

$$0 \leq z_i \leq W \qquad \forall i \in \mathcal{N}' \tag{3.19}$$

$$z_j = z_i - y_{ij} \qquad \forall (i,j) \in \mathcal{A}, j \in \mathcal{S} \cup \mathcal{C}^+ \tag{3.20}$$

$$z_j = z_i + y_{ij} \qquad \forall (i,j) \in \mathcal{A}, j \in \mathcal{D} \cup \mathcal{C}' \tag{3.21}$$

$$z_i \in \mathbb{Z}+ \qquad \forall i \in \mathcal{N}' \tag{3.22}$$

Constraint (3.9) makes sure that K shuttles leave depot start node 0 and enter depot end node $N+1$. Constraint (3.10) ensures the flow balance between intermediate nodes. Constraint (3.11) eliminates subtours. Except for the depot node, supply nodes and charge stations can be visited once or not (Constraint (3.12)), and demand nodes must be visited once (Constraint (3.13)). If a charge node $c \in \mathcal{C}'$ is visited by a shuttle, its paired dummy charge process node $c^+ \in \mathcal{C}^+$ must be visited by a shuttle (Constraint (3.14)). Constraint (3.16) restricts the times when a shuttle arrives at node j. All shuttles leave the depot with W workers (Constraint (3.18)). The number of workers on a shuttle must be nonnegative and less than or equal to the capacity W (Constraint (3.19)). Constraints (3.20) and (3.21) show that one shuttle drops off one worker on a supply node or a dummy charge process node and picks up one worker on a demand node or a charge node. Constraints (3.15), (3.17), and (3.22) restrict that routing decision y is binary variable, shuttle arrival time τ is nonnegative continuous variable and the number of drivers onboard shuttles is integer variable.

3.3.3 Synchronizing EV Relocation and Shuttles Routing Decisions

The following part is to synchronize EV relocation and shuttle routing problems.

$$\sum_{(i,s)\in\mathcal{A}} y_{is} = \sum_{p\in\phi(s)} x_p \qquad \forall s \in \mathcal{S} \tag{3.23}$$

$$\sum_{(i,c)\in\mathcal{A}} y_{ic} \leq \sum_{p\in\phi(c)} x_p \qquad \forall c \in \mathcal{C}' \tag{3.24}$$

$$e_s \geq \tau_s - M(1 - \sum_{p\in\phi(s)} x_p) \qquad \forall s \in \mathcal{S} \tag{3.25}$$

$$\tau_c \geq e_c - M(1 - \sum_{(i,c)\in\mathcal{A}} y_{ic}) \qquad \forall c \in \mathcal{C}' \tag{3.26}$$

$$e_{c^+} \geq \tau_{c^+} - M(1 - \sum_{(i,c^+)\in\mathcal{A}} y_{ic^+}) \qquad \forall c^+ \in \mathcal{C}^+ \tag{3.27}$$

$$\tau_d \geq e_d \qquad \forall d \in \mathcal{D} \tag{3.28}$$

Constraint (3.23) shows that when an EV at node s is assigned to relocate to node d, the shuttle must pass through node s. Constraint (3.24) shows that the shuttle passes through charge station c (the worker is picked by this shuttle) or does not pass charge station node c (the worker chooses to wait for EV to be charged). The shuttles routes restrict EVs' time window. Assume the times of getting on and off shuttles are not considered. When a shuttle drop off a worker at supply s or a dummy charge process node $c^+ \in \mathcal{C}^+$, EV starts to leave (Constraints (3.25) and (3.27)). Constraints (3.26) and (3.28) show that the shuttle picks up a driver and leaves charge station c or demand d after EV has already arrived.

The mathematical model for the FFEVSS is formulated as follows.

minimize	τ_{N+1}	
subject to	$(3.1) - (3.8)$	[EV Relocation]
	$(3.9) - (3.22)$	[Shuttle Routing]
	$(3.23) - (3.28)$	[Synchronize both]

3.4 Benchmark Methods

In this section, two benchmark methods, exchange-based neighborhood-search method (EBNSM) (Haider et al., 2019) and reinforcement learning approach (RL) (Bogyrbayeva et al., 2021) are introduced in brief.

3.4.1 Exchange-Based Neighborhood-Search Method

Haider et al. (2019) create a sequential approach. In the first stage, it gets the best EV relocation paths. In the second stage, given the best EV relocation paths, the initial shuttles routes are obtained by a greedy approach. Given the initial shuttles routes, a 2-interchange method is used to get the best shuttles routes. Then, EBNSM improves the

initial solution that is obtained from the sequential method by iteratively creating EV paths and updating the shuttle route. Exchange procedures of suppliers and chargers are added to update shuttles routes. When a pair of old EV relocation paths are replaced by a new pair, their supplier and charger nodes are exchanged, respectively. The route update step swaps the positions of the pair of exchanged nodes in the shuttle route based on the new EV relocation path. Meanwhile, the precedence feasibility of visiting nodes and the capacity feasibility of shuttles are also maintained.

The main difference between EBNSM and ALNS is that in EBNSM, each charge station is allowed to visit exactly once, while ALNS allows them to visit multiple times. Thus, EBNSM enables to get a feasible solution based on requesting that the number of available charge stations is more than the number of pairs of EV relocation which requires a charge. However, in reality, the number of charge stations is probably less than the number of EVs which require a charge. It is realistic for EVs to wait in a queue at one charge station. Moreover, the time window is also a variant value as the pairs of EV relocation and shuttles routes change. So, another heuristic method is necessary to develop with consideration of multiple visiting of charge stations.

3.4.2 Reinforcement Learning Method

Bogyrbayeva et al. (2021) formulate a reinforcement learning framework and deploy a policy gradient method for training recurrent neural networks. The problem is formulated as a finite horizon Markov Decision Process with the state, the action, the transition probabilities, and the reward function. The state represents the network that shows each node location, the distance, the number of EVs, the number of drivers, the battery level, and indicators for the expected transitions. The action indicates a node number to be visited next by which shuttle. The reward is defined as the negative of the total time spent in the system that is measured from the time when shuttles leave a depot to the time when all shuttles return. The reward function is to maximize the total expected reward. A simu-

lator is proposed to describe the dynamics caused by shuttles routes. This simulator also ensures the precedence feasibility of visiting nodes and the capacity feasibility of drivers on a shuttle by following a masking scheme. Bogyrbayeva et al. (2021) focus on solving the shuttle routing and making a decision on EV relocation by using the nearest rule. When an EV at the supplier has a driver, the EV is relocated to the nearest available charge station or demand node. On average computational time, RL outperforms EBNSM, but requires lengthy training.

The main difference between RL and ALNS is that ALNS can search more further neighborhoods of both EV relocation and shuttles routes, while RL only uses the nearest location rule to find EV relocation decisions. Thus, ALNS may obtain a better solution than RL.

3.5 Adaptive Large Neighborhood Search

In this section, we develop an Adaptive Large Neighborhood Search (ALNS) for the free-floating EV sharing system. Ropke and Pisinger (2006) developed ALNS as an extension of the Large Neighborhood Search method (Shaw, 1998). The idea of ALNS is to search in a large neighborhood using multiple destroy and repair methods and to choose the destroy and repair methods based on their adaptive probabilities.

To the best of our knowledge, this is the first paper using ALNS on the free-floating carsharing system problem that considers EV relocation and shuttle routing jointly. In the original ALNS (Ropke and Pisinger, 2006), the feasible solution only has one decision (request routes). However, in this paper, the feasible solution has two decisions: EV relocation X and shuttle routes Y. The main challenge is to jump from one solution to a new solution in the neighborhood. Because two decisions affect each other, the processes of destroy and repair become more complicated. So, the repair methods are divided into two stages. In the first stage, greedy and probabilistic methods are proposed to match suppliers and demanders to repair EV relocations. In the second stage, greedy and regret methods are presented to reconstruct shuttle routes.

Algorithm 1: Pseudocode for ALNS

Input: $\mathcal{D}$, $\mathcal{S}$, $\mathcal{C}$, DM, RM, $N_{\max}$, $Z_{\max}$
Output: X_{best}, Y_{best}

Initialize EV relocation X_0 and shuttle routes Y_0 (Sec 3.5.1);
Initialize destroy methods probability $\mathbf{P}_D^0$ and repair methods probability $\mathbf{P}_R^0$ (Sec 3.5.4);
$X_{\text{best}} \leftarrow X_{\text{current}} \leftarrow X_0$, $Y_{\text{best}} \leftarrow Y_{\text{current}} \leftarrow Y_0$;
Calculate the makespan of current best solution $t_{\text{best}} \leftarrow f(X_{\text{best}}, Y_{\text{best}})$;
$N \leftarrow 1, Z \leftarrow 0$;
while $N \leq N_{\max}$, $Z \leq Z_{\max}$ **do**
    Select a destroy method $d \in$ DM with probability P_D^N;
    Select a repair method $r \in$ RM with probability P_R^N;
    Let X_{new} and Y_{new} be the new solution obtained by apply destroy d and repair r;
    if $f(X_{\text{new}}, Y_{\text{new}}) < t_{\text{best}}$ **then**
        $X_{\text{best}} \leftarrow X_{\text{new}}$, $Y_{\text{best}} \leftarrow Y_{\text{new}}$, $t_{\text{best}} \leftarrow f(X_{\text{new}}, Y_{\text{new}})$, $Z \leftarrow 0$;
    else
        $Z \leftarrow Z + 1$;
        $v = e^{-(f(X_{\text{new}}, Y_{\text{new}}) - f(X_{\text{current}}, Y_{\text{current}}))/T}$;
        Generate a random number $\epsilon \in [0, 1]$;
        if $\epsilon < v$ **then**
            $X_{\text{current}} \leftarrow X_{\text{new}}$, $Y_{\text{current}} \leftarrow Y_{\text{new}}$;
    $T \leftarrow hT$;
    Update $\mathbf{P}_D^N$ and $\mathbf{P}_R^N$ (Sec 3.5.4);
    $N \leftarrow N + 1$;

The procedure of the proposed ALNS is shown in Algorithm 1. In each iteration, the neighborhood of a solution is produced by destroy and repair. We let DM and RM denote the sets of the destroy and repair methods, respectively. The destroy process is to remove part of EV relocation solution X and shuttle routes Y. Next, the repair process can reconstruct the partially incomplete solution. The destroy and repair methods are chosen according to their past successes. When a better solution is obtained by applying the methods, the counts of the corresponding methods are added by one, and their probabilities will increase, as described in detail in section 3.5.4. The roulette-wheel selection principle is used for the selection of destroy and repair. In this paper, a simple acceptance rule is applied: the new best solution is accepted if its objective value is better than the current best solution. The new solution with a higher objective value is accepted by a simulated annealing acceptance

criterion. T denotes the value of the temperature and gradually decreases at each iteration by a rate $h \in [0, 1]$. The stop criteria are the maximum iterations $N_{\max}$ and non-improving iteration $Z_{\max}$.

3.5.1 Finding an Initial Solution

The nearest neighborhood (NNH) search (Algorithm 2) is applied to generate an initial solution as follows. We randomly select a demand node d in undecided demand node-set UD and match its nearest supply node s in undecided supply node-set US. If EV at the node s requires being charged, then insert the nearest charge station c in the middle. Then, delete s and d in the undecided set US and UD. The EV relocation ω is assigned on the shuttles in sequence.

Algorithm 2: Pseudocode for NNH

Input: $\mathcal{D}, \mathcal{S}, \mathcal{S}^c, \mathcal{C}$
Output: X, Y
Initialize EV relocation set: $X_1 \leftarrow \emptyset$, $X_2 \leftarrow \emptyset$, $X \leftarrow X_1 \cup X_2$;
Initialize shuttle routes: $Y_k \leftarrow \emptyset \ \forall k \in \mathcal{K}$, $Y \leftarrow \cup_{k \in \mathcal{K}} Y_k$;
Initialize search sets: $\mathsf{UD} \leftarrow \mathcal{D}$, $\mathsf{US} \leftarrow \mathcal{S}$;
$k \leftarrow 1$;
while $\mathsf{UD} \neq \emptyset$ **do**
    $\forall d \in \mathsf{UD}$, find the nearest supplier $s = \arg\min\{d_{sd}, \forall s \in \mathsf{US}\}$;
    if $s \in \mathcal{S}^c$, **then**
        Create a EV relocation $\omega_{sd} \leftarrow \{s \rightarrow c \rightarrow d\}$, where $c = \arg\min\{d_{sc} + d_{cd}, \forall c \in \mathcal{C}\}$;
        $X_2 \leftarrow X_2 \cup \{\omega_{sd}\}$;
    else
        Create a EV relocation $\omega_{sd} \leftarrow \{s \rightarrow d\}$;
        $X_1 \leftarrow X_1 \cup \{\omega_{sd}\}$;
    $\mathsf{US} \leftarrow \mathsf{US} \setminus \{s\}$, $\mathsf{UD} \leftarrow \mathsf{UD} \setminus \{d\}$;
    Add s and d to the end of kth shuttle route list Y_k;
    if $k = K$ **then**
        $k \leftarrow 1$;
    else
        $k \leftarrow k + 1$;

An initial solution consists of two decisions: EV relocation denoted as X and shuttle routes denoted as Y. EV relocation decision X comprises of two distinct types: (1) an EV directly moves from a supply node to a demand node, and (2) an EV moves to a charge station and then goes to a demand node. Shuttle routes serve these EV relocation operations, and the corresponding shuttle route set Y consists of K lists, namely $Y_1, Y_2, ..., Y_K$. The list Y_k means the sequence nodes visited by the k-th shuttle.

3.5.2 Destroy Methods

Three destroy methods are applied to destroy a complete feasible solution into a partial one. The destroy methods consist of Random Removal, Worst Route Removal, and Cluster Removal, are presented as follows.

3.5.2.1 Random Removal

Randomly remove $\lfloor \alpha\% \times |D| \rfloor$ EV relocation ω_{sd} in X, where α is the destroy percentage. Put these supply nodes and demand nodes into the undecided supplier set (US) and undecided demand set (UD). Meanwhile, delete them in shuttle routes Y.

3.5.2.2 Worst Route Removal

Given a solution (X, Y), a demand d is matched with the supply s in EV relocation solution X. we define the cost for EV relocation pair ω_{sd} as

$$\text{cost}(\omega_{sd}, X, Y) = f(X, Y) - f_{-\omega_{sd}}(X, Y) \tag{3.29}$$

where $f_{-\omega_{sd}}(X, Y)$ is the objective value without EV relocation pair ω_{sd} in (X, Y). It is reasonable to remove the supply-demand pair with the high cost and perhaps create new EV relocation to obtain a better solution. Sort all supply-demand pairs ω in descending costs. Remove first $\lfloor \alpha\% \times |D| \rfloor$ EV relocation ω with larger costs. Put these supply nodes

and demand nodes into the undecided supplier set (US) and undecided demand set (UD). Meanwhile, delete them from shuttle routes Y.

3.5.2.3 Cluster Removal

The idea of cluster removal is to remove the similar demand nodes (Shaw, 1998). Since a new and better solution is expected to be created, the current solution needs to be destroyed more heavily. This allows the farther neighborhood to be searched.

The relatedness between node i and j is used to measure how node j is close to node i. For any demand node i, the measure of relatedness is calculated as the following equation.

$$R(i,j) = w_1 \frac{d_{ij}}{\max\{d_{ik}, \forall k \in \mathcal{D}\}} + w_2 \frac{|e_i - e_j|}{\max\{e_k, \forall k \in \mathcal{D}\} - \min\{e_k, \forall k \in \mathcal{D}\}} \quad \forall j \neq i \in \mathcal{D} \tag{3.30}$$

where w_1 and w_2 are weights with sum of 1. d_{ij} is the distance between node i and j. e_i is the time when an EV arrives at demand node i. The smaller $R(i,j)$ is, the more related the demands i and j are.

The following steps are to randomly select a demand node $d \in \mathcal{D}$ and calculate $R(d,j), \forall j \neq d \in \mathcal{D}$. Sort all $R(d,j)$ in descending order. Remove demands with the first $\lfloor \alpha\% \times |D| \rfloor$ in the sequence and their corresponding supply nodes. Put these supply nodes and demand nodes into undecided supplier (US) and undecided demand sets (UD). Meanwhile, delete them from shuttle routes Y.

3.5.3 Repair Methods

The following repair process must conform to two constraints:

- Sequence constraint: supply s must be visited before demand d, when supply s is paired with demand d in EV relocation. Charge station c must be visited between s and d. Its paired dummy charger c^+ must be visited after c.

- Personnel constraint: Since there are a limited number of workers (capacity) onboard each shuttle, in any shuttle route Y_k, the difference between the numbers of visiting supply and demand nodes must stay within $[0, W]$. For example, given a shuttle route $Y_k = \{a_1, a_2, a_3, ...\}$.

  For each positive integer $n = 1, 2, ..., |Y_k|$, we enforce

$$0 \leq \sum_{i=1}^{n} \mathbb{1}[a_i \in \mathcal{S} \cup \mathcal{C}^+] - \sum_{i=1}^{n} \mathbb{1}[a_i \in \mathcal{D} \cup \mathcal{C}] \leq W$$

where $\mathbb{1}[\cdot]$ equals one if the condition inside the bracket holds and equals zero otherwise.

We have two repair rules for EV relocation and two repair rules for routes insertion. Four repair methods are formed by combining these repair rules.

- Sequential Greedy: Greedy s-d matching with Greedy routes insertion
- Sequential Regret: Greedy s-d matching with Regret routes insertion
- Probabilistic Greedy: Probabilistic s-d matching with Greedy routes insertion
- Probabilistic Regret: Probabilistic s-d matching with Regret routes insertion

3.5.3.1 Repair Rules for EV Relocation

Two repair rules for EV relocation are applied to match a supply node with a demand node, and are described as follows:

- Greedy s-d matching: For $\forall d \in \mathsf{UD}$, demand d is matched with an undecided supply s where $s = \text{argmin}\{d_{sd}, s \in \mathsf{US}\}$. If $s \in \mathcal{S}^c$, $\omega_{sd} = \{s \to c \to d\}$, where $c = \text{argmin}\{d_{sc} + d_{cd}, c \in \mathcal{C}\}$; else $\omega_{sd} = \{s \to d\}$. The new supply-demand pair ω_{sd} is formed by the nearest-neighbor rule.

- Probabilistic s-d matching: s and d are matched as a new pair based on the probability that is related to their distance. The s-d matching probability is expressed as

$$P(\omega_{sd}) = \frac{\exp(-\lambda d_{\omega_{sd}})}{\sum_{\omega_{sd} \in \mathcal{U}} \exp(-\lambda d_{\omega_{sd}})} \tag{3.31}$$

where $\mathcal{U}$ is the set of all combinations of undecided $s \in \mathsf{US}$ and undecided $d \in \mathsf{UD}$. λ is a constant within $[0, 1]$. If $s \in \mathcal{S}^c$, each charge station is inserted in the middle of s and d. There are $|\mathcal{C}|$ combinations for s and d. For each $\omega_{sd} = \{s \to c \to d\}$, we set $d_{\omega_{sd}} = d_{sc} + d_{cd}$. If $s \in \mathcal{S}^n$, we set $d_{\omega_{sd}} = d_{sd}$.

3.5.3.2 Repair Rules for Routes Insertion

Two repair rules for routes insertion are applied to insert supply-demand pair into the shuttle routes, and are described as follows.

- Greedy routes insertion: A concept of Insertion Cost is introduced. Let $\Delta f(n, p, Y)$ denote the change in the objective value when inserting node n into partial shuttle routes Y at position p. The insertion cost is expressed as

$$I(n, p, Y) = \Delta f(n, p, Y) \tag{3.32}$$

For any new supply-demand ω_{sd}, insert s and d into the current partial shuttle routes Y, separately. Select the position p to insert s with the least insertion cost $I(s, p, Y)$. The new partial shuttle routes are formed after inserting s, denoted as Y_{+s}. Then, insert d at the position with the least insertion cost $I(d, p, Y_{+s})$ and get the new partial routes Y. If $s \in \mathcal{S}^c$, insert c and c^+ with the least insertion cost sequentially. Repeat the above steps until all demands are satisfied.

- Regret routes insertion: The regret routes insertion is improved by incorporating look-ahead information when selecting the supply-demand pair to insert.

For any supply-demand pair ω_{sd}, the regret-k cost is defined as

$$R(\omega_{sd}, Y) = \sum_{j=1}^{k} \{\Delta f_j(\omega_{sd}, Y) - \Delta f_1(\omega_{sd}, Y)\} \tag{3.33}$$

where $\Delta f(\omega_{sd}, Y)$ is the increased value in the objective value after inserting s and d. Sort $\Delta f(\omega_{sd}, Y)$ for all possible insertion positions in the increasing order. The best insertion position has the least $\Delta f_1(\omega_{sd}, Y)$. Note that $\Delta f_k(\omega_{sd}, Y)$ means the increased value in the objective for the k-th best insertion position. The regret routes insertion is the reconstruction heuristic that chooses to insert the supply-demand pair ω_{sd} with the maximum $R(\omega_{sd}, Y)$. The ω_{sd} is inserted at its minimum cost position. If $s \in \mathcal{S}^c$, insert c and c^+ with the least insertion cost sequentially. Repeat the above steps until all demands are satisfied.

3.5.4 Adaptive Probability Update Procedure

The adaptivity of ALNS is achieved by selecting the destroy and repair methods based on their previous successes. The initial probabilities of destroys and repairs are set to 1 divided by the number of available destroy and repair methods; that is, $\frac{1}{|\mathrm{DM}|}$ and $\frac{1}{|\mathrm{RM}|}$. In each iteration i, if destroy method d and repair method r create a new best solution X_{best}, the count n_d^i and n_r^i of destroy d and repair r is increased by 1, respectively. Then, the probability values of destroy and repair methods in iteration N are updated by multiplication of two sets of ratios σ_d and σ_r as follows:

$$\mathbf{P}_D^N = \left(\frac{\sigma_d n_d^N}{\sum_{d \in \mathrm{DM}} \sigma_d n_d^N} : d \in \mathrm{DM} \right) \tag{3.34}$$

$$\mathbf{P}_R^N = \left(\frac{\sigma_r n_r^N}{\sum_{r \in \mathrm{RM}} \sigma_r n_r^N} : r \in \mathrm{RM} \right) \tag{3.35}$$

The destroy and repair methods are adaptively chosen based on the new probabilities by using the roulette-wheel selection principle.

3.6 Modification of ALNS in Problem Variants

We consider two variants of the rebalancing problem. First, we consider the case when EV drivers use their own personal mobility option instead of shuttles. Second, we consider the dynamic environments wherein EV supply or demand locations change while shuttles and drivers are already executing an operational plan. In both cases, we show that ALNS can be easily modified.

3.6.1 Routing with Personal Mobility Options

In some cases, the workers can be moved not only by shuttles but also by personal mobility vehicles such as scooters. Each worker has one personal mobility vehicle. When a worker arrives at a supply node, this worker puts the scooter in the back and drives the EV to the demand node or a charge station. When the EV arrives at a charge station, the worker can ride his mobility tool to another node to accomplish other tasks. In the sequence constraint, $s \to c$ and $c \to d$ are bound as one unit. Also, $s \to c$ must be inserted before $c \to d$ in the routes. The personnel constraint is necessary to be taken into account.

EV relocation is same as stated in Section 3.5. It has two types: $\omega_{sd} = \{s \to d\}$ and $\omega_{sd} = \{s \to c \to d\}$. The destroy and repair methods for EV relocation do not change.

The repair rules for routes insertion need some changes as follows. Because the scooters and workers move synchronously with the EVs, the worker routing is consistent with EV relocation from $s \to d$, $s \to c$ and $c \to d$. We can regard $s \to d$, $s \to c$ and $c \to d$ as one unit and insert them into the partial routes. For an example, some EV relocations are $s1 \to d4$, $s2 \to c3$, and $c5^+ \to d6$. One worker route can be described as $(1 \to 4) \to (2 \to 3) \to (5^+ \to 6)$. That means that this worker drives an EV from supply 1 to demand 4 and then goes to node 2 by riding his scooter; he drives the EV from supply 2 to charge station 3; Finally, he rides the scooter to dummy charge station 5^+ and drives the EV to demand 6.

3.6.2 EV Relocation and Routing in Dynamic Environments

In dynamic environment, the number of EV relocation assignments can change in the middle of the relocation operations. It happens in the cases when some of the current available EVs are assigned to the arriving customers or some additional demands are added into the system. Instead of solving the new routing problem from scratch, we can use the ALNS algorithm to destroy and repair the current solution to adapt to the dynamic environment. When we repair, we just ignore those EVs which have been already served. Remove the decreased demands (or suppliers) or add the additional demands (or suppliers) to the undecided demand set UD (or the undecided supply set US).

The shuttles have departed the depot, and some EV relocation demands $\widetilde{\mathcal{D}}$ have been served already. The changes in demand and supply sets are described as sets themselves and denoted as $\mathcal{D}^*$ and $\mathcal{S}^*$. There are two cases.

- When the number of EV demands decreases: The decreased demand nodes and corresponding supply nodes are removed from the current solution. Then, partial EV relocation X and partial shuttle routes Y are obtained. This process is just like a destroy operation. The next step is to repair the current partial sets X and Y by four repair methods in Section 3.5.3.

- When additional EV demands are added in the middle of relocation operations: The additional EV demands and supply sets $\mathcal{D}^*$ and $\mathcal{S}^*$ are added to the undecided demand set UD, and the undecided supply set US. Then a new solution is obtained after repairing based on the new demand and supply nodes.

The process to create a new solution in the dynamic environment is described in Algorithm 3. We let X^0 and Y^0 denote current EV relocation and shuttle routing solutions, respectively. $\widetilde{\mathcal{D}}$ and $\widetilde{\mathcal{S}}$ are defined as the finished demands and suppliers, respectively.

Algorithm 3: Pseudocode for ALNS in Dynamic Environment

Input: $\mathcal{D}, \widetilde{\mathcal{D}}, \mathcal{D}^*, \mathcal{S}, \widetilde{\mathcal{S}}, \mathcal{S}^*, X^0, Y^0$
Output: $X_{\text{best}}, Y_{\text{best}}$

The partial EV relocation $X \leftarrow X^0 \setminus \{\omega_{sd}, \forall \omega_{sd} \in X^0 : d \in \widetilde{\mathcal{D}}\}$;
The partial shuttle routing $Y \leftarrow Y^0 \setminus \{s, d, \forall s \in \widetilde{\mathcal{S}}, d \in \widetilde{\mathcal{D}}\}$;
Initialize undecided demand set $\mathsf{UD} \leftarrow \mathcal{D} \setminus \widetilde{\mathcal{D}} \cup \mathcal{D}^*$ and undecided supply set $\mathsf{US} \leftarrow \mathcal{S} \setminus \widetilde{\mathcal{S}} \cup \mathcal{S}^*$;
Initialize destroy methods probability $\mathbf{P}_D^0$ and repair methods probability $\mathbf{P}_R^0$ (Sec 3.5.4);
Apply a repair method $r \in \mathsf{RM}$ with probability P_R^0 on X, Y and $X_{\text{new}}, Y_{\text{new}}$ are obtained;
$X_{\text{best}} \leftarrow X_{\text{current}} \leftarrow X_{\text{new}}$, $Y_{\text{best}} \leftarrow Y_{\text{current}} \leftarrow Y_{\text{new}}$;
Calculate the makespan of current best solution $t_{\text{best}} \leftarrow f'(X_{\text{best}}, Y_{\text{best}}, \Psi)$;
$N \leftarrow 1, Z \leftarrow 0$;
while $N \leq N_{\mathsf{max}}$, $Z \leq Z_{\mathsf{max}}$ **do**
    Select a destroy method $d \in \mathsf{DM}$ with probability P_D^N;
    Select a repair method $r \in \mathsf{RM}$ with probability P_R^N;
    Let X_{new} and Y_{new} be the new solution obtained by apply destroy d and repair r;
    if $f'(X_{\mathsf{new}}, Y_{\mathsf{new}}, \Psi) < t_{\mathsf{best}}$ **then**
        $X_{\mathsf{best}} \leftarrow X_{\mathsf{new}}$, $Y_{\mathsf{best}} \leftarrow Y_{\mathsf{new}}$, $t_{\mathsf{best}} \leftarrow f'(X_{\mathsf{new}}, Y_{\mathsf{new}}, \Psi)$, $Z \leftarrow 0$;
    else
        $Z \leftarrow Z + 1$;
        $v = e^{-(f'(X_{\mathsf{new}}, Y_{\mathsf{new}}, \Psi) - f'(X_{\mathsf{current}}, Y_{\mathsf{current}}, \Psi))/T}$;
        Generate a random number $\epsilon \in [0, 1]$;
        if $\epsilon < v$ **then**
            $X_{\text{current}} \leftarrow X_{\text{new}}$, $Y_{\text{current}} \leftarrow Y_{\text{new}}$;
    $T \leftarrow hT$;
    Update $\mathbf{P}_D^N$ and $\mathbf{P}_R^N$ (Section 3.5.4);
    $N \leftarrow N + 1$;

Let the tuple of lists

$$\Psi = \left(\left[(e_i, \tau_i) : i \in \widetilde{\mathcal{D}} \cup \widetilde{\mathcal{S}} \right], \left[\mathcal{L}(k) : k \in \mathcal{K} \right], \left[\mathcal{O}(k) : k \in \mathcal{K} \right] \right)$$

denote the current state of the system, which includes EVs' and shuttles' arrival time e_i, τ_i at node i; the locations $\mathcal{L}(k)$ of shuttle k; and the number of workers $\mathcal{O}(k)$ onboard shuttle k. Because the finished EV relocation and shuttle routing cannot be changed, the follow-

ing operations are done to reorganize the unfinished demands and suppliers. The value of makespan is calculated based on the current state of the system Ψ.

3.7 Numerical Experiments

The experiments are done on a computer that runs 64-bit Windows 10 with a 2.60 GHz Intel Core (TM i5-7300U) CPU and 8 GB RAM. ALNS and EBNSM are coded in Julia 1.6.1. The Reinforcement Learning approach of Bogyrbayeva et al. (2021) is implemented in Python 3.6.

3.7.1 Randomly Generated Instances

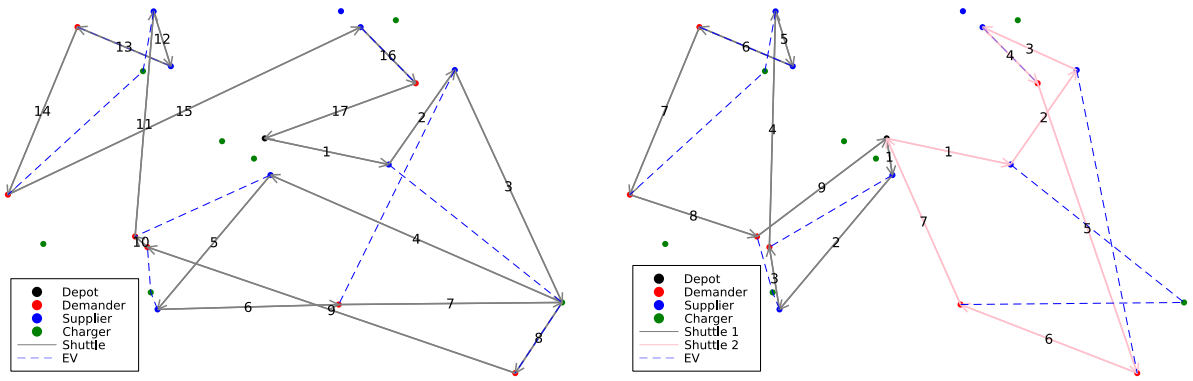
We consider a 10×10 miles square network, which is proximate to one urban city area. There are supplier, charger, and demand nodes within the network. We fix the total number of nodes and the number of suppliers, chargers, and demand nodes. The x and y coordinates of each node are generated by a uniform distribution from 0 to 10. The initial residual charge level of the battery for each supplier node is generated randomly between 0 to 100%. Assume that the speed of each EV and the speed of each shuttle are equal to 65mph and 40mph, respectively.

We assume that EVs do not need to be charged and directly move to one demand node if the initial battery level is more than 50%. Otherwise, the EV moves to a charge station at first and is required to be charged fully. The charge time is assumed as 40 minutes from empty to full. The charging energy is directly proportional to the time. For example, an EV with 40% remaining battery level will spend 24 minutes to be charged fully. We do not consider the battery consumption of EV movement. In ALNS, the stop criteria are set as the maximum iteration $N_{\max}$ of 3000 and Maximum non-improving iteration $Z_{\max}$ of 300.

The test instances are generated by three difficulty types. When the number of charge stations is more than the number of suppliers requiring recharged, the instance is easy; When they are the same, it is medium; otherwise, it is hard. The number of nodes $|\mathcal{N}|$, the number

Table 3.2: Types of Random Instances

Nodes Num	Easy				Medium				Hard			
$\lvert\mathcal{N}\rvert$	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{S}^c\rvert$	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{S}^c\rvert$	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{S}^c\rvert$
23	7	7	8	4	7	4	8	4	7	2	8	4
50	17	15	17	10	17	10	17	10	17	5	17	10
150	50	50	50	30	50	30	50	30	50	20	50	30



(a) single-shuttle case, $K = 1$

(b) multi-shuttle case, $K = 2$

Figure 3.3: Example Solutions for $|\mathcal{N}| = 23$ and $W = 3$

of demand nodes $|\mathcal{D}|$, the number of charge stations $|\mathcal{C}|$, the number of supply nodes $|\mathcal{S}|$, and the number of supply nodes requiring charging $|\mathcal{S}^c|$ are summarized in Table 3.2.

Parameters affect the performance of the algorithm. After testing these parameters on the randomly generated instances, they are tuned as follows. 1) destroy percentage $\alpha\% = 30\%$; 2) the constant in Probabilistic s-d matching $\lambda = 0.5$; and 3) ratios for destroy $\sigma_d = (0.35, 0.4, 0.25)$ and ratios for repair methods $\sigma_r = (0.3, 0.2, 0.3, 0.2)$.

ALNS, EBNSM, and RL are implemented on 10 instances for each type. The example solutions for single shuttle and multiple shuttles are shown in Figure 3.3a and Figure 3.3b, respectively. The solutions are obtained by using ALNS to solve one easy instance with $|\mathcal{N}| = 23$.

Table 3.3: Average Objective Values of ALNS, EBNSM, and RL on Random Instances

Instances			Easy				Medium				Hard			
$\|\mathcal{N}\|$	K	W	EBNSM	RL	ALNS	PD (%)	EBNSM	RL	ALNS	PD (%)	EBNSM	RL	ALNS	PD (%)
23	1	3	8.81	7.70	**7.52**	2.39	12.39	10.27	**10.10**	1.68	-	12.32	**11.95**	3.10
	2	3	5.72	5.40	**5.11**	5.68	7.43	6.40	**6.03**	6.14	-	8.34	**8.14**	2.46
	3	2	5.27	5.21	**4.87**	6.98	6.39	6.38	**6.01**	6.16	-	7.79	**7.11**	9.56
50	1	3	17.34	13.77	**12.33**	11.68	24.49	17.93	**15.78**	13.62	-	18.92	**17.18**	10.13
	2	3	9.19	8.41	**8.20**	2.56	12.25	11.23	**10.34**	8.61	-	11.96	**10.10**	18.42
	3	2	6.96	5.89	**5.78**	1.90	9.25	9.23	**9.05**	1.99	-	9.77	**9.48**	3.06
150	1	3	34.30	22.18	**21.31**	4.08	45.97	30.67	**27.75**	10.52	-	36.67	**32.52**	12.76
	2	3	16.11	12.92	**11.67**	10.71	21.63	17.54	**17.15**	2.27	-	17.90	**16.77**	6.74
	3	2	11.71	10.21	**9.28**	10.02	15.63	13.33	**12.50**	6.64	-	14.94	**13.05**	14.48
Average						6.22				6.40				8.97

The averages of objective values are summarized in Table 3.3. The percentage deviations between RL and ALNS are calculated as follows.

$$\text{PD} = \frac{\text{Objective of RL} - \text{Objective of ALNS}}{\text{Objective of ALNS}} \times 100\% \tag{3.36}$$

For all instances, ALNS can obtain the best average objective values. In the EBNSM structure, the charge stations are not allowed to be visited more than one time, so a feasible solution cannot be found in the hard type instances. The average percentage deviations between RL and ALNS for easy, medium, and hard instances are 6.22, 6.40, and 8.97, respectively. Therefore, ALNS can perform better than both EBNSM and RL in the solution quality.

The average computational times are shown in Table 3.4. After training on the data set, the trained RL model can take less than 1 second to get the solution. The training time is not included in the computational times in Table 3.4. So, the computational times of RL cannot be directly compared with the computational times of ALNS. We also observe that ALNS can run faster 21.6% and 19.7% than EBNSM for easy and medium instances.

The numerical experiments on randomly generated instances demonstrate that ALNS outperforms both EBNSM and RL in terms of the solution quality in all problem sizes. The computational times of ALNS are shorter than EBNSM, but much longer than RL. While

Table 3.4: Average Computational Times of ALNS, EBNSM, and RL on Random Instances (Unit: Second)

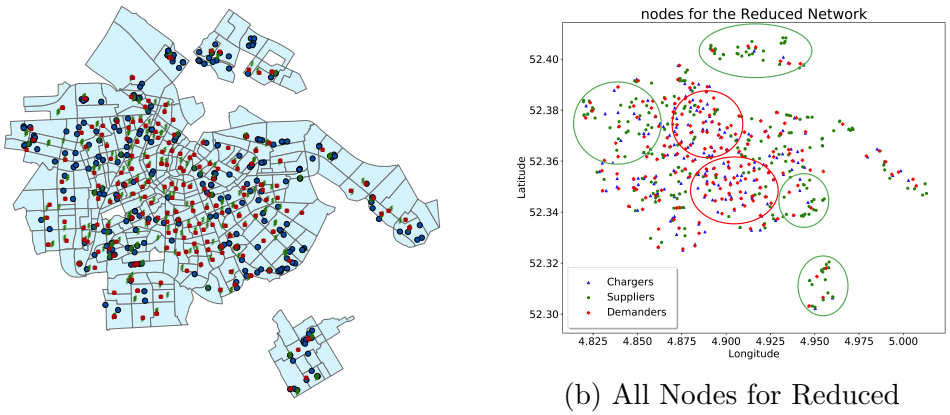
Instances			Easy			Medium			Hard		
$\|\mathcal{N}\|$	K	W	RL	EBNSM	ALNS	RL	EBNSM	ALNS	RL	EBNSM	ALNS
23	1	3	0.01	6.83	**6.57**	0.02	10.38	**6.32**	0.02	-	12.32
	2	3	0.04	3.76	**1.61**	0.04	3.52	**3.31**	0.09	-	9.52
	3	2	0.06	9.17	**7.11**	0.05	4.56	**3.65**	0.11	-	7.21
50	1	3	0.05	40.42	**35.23**	0.05	58.59	**45.10**	0.06	-	53.42
	2	3	0.21	29.57	**21.52**	0.16	35.17	**28.74**	0.25	-	29.23
	3	2	0.20	20.20	**15.82**	0.21	25.20	**18.75**	0.35	-	54.18
150	1	3	0.16	99.25	**76.72**	0.17	105.20	**89.24**	0.26	-	111.23
	2	3	0.44	87.42	**71.23**	0.44	77.31	**60.24**	0.55	-	97.21
	3	2	0.92	142.45	**125.24**	1.03	183.42	**165.18**	1.02	-	152.42
Average			0.21	48.79	40.12	0.24	55.93	46.73	0.30	-	58.53

RL executes in less than a second in most cases, RL requires very long training time; several days to a couple of weeks depending on the hardware used. On the other hand, ALNS can be applied to each problem directly without lengthy training.

3.7.2 Case Study: Car2go in Amsterdam

We apply our approach to a fully operational system of car2go in Amsterdam, the Netherlands, where the FFEVS service is operational using more than 300 EVs. From the actual data, we take the initial and target locations of EVs that need to be relocated and test the performance of our computational method. The nodes are depicted in Figure 3.4-(b), which clearly shows that the current EV locations (suppliers) and the desired locations (demanders) are concentrated in different areas, hence necessitating relocation operations.

To create a case study, we sample test data from 12:00 am to 7:00 am on each day. Within this period, if one EV moves from one place s to another place d, place s is set as a supply node, and place d is set as a demand node. When one EV stays at one location and the battery level increases, the location c is set as a charge station. The 143 days data are sampled, and their dates range from May to October in 2016. The numbers of suppliers,



(a) Map for the Neighborhoods

(b) All Nodes for Reduced Network

Figure 3.4: Amsterdam Network. Left: Map for the Neighborhoods together with Suppliers (blue circle), Demanders (red square), Chargers (green bolt sign) and Depot node (Green Pentagon); Right: All Nodes for Reduced Network

chargers and demanders in the final network stay within the range of [46, 251], [21, 64] and [46, 251]. The average numbers of suppliers, chargers and demanders are 85.3, 39.2, and 85.3, respectively. Since the default speed limit in the city of Amsterdam is 50 km/h inside built-up areas, the speed of an EV is assumed as 50 km/h. In urban residential areas, 30 km/h zones are found on the living streets. We assume the shuttles can move within these areas, so the speed of a shuttle is assumed as 30 km/h.

Two scenarios are considered in this section:

- Scenario 1: No parking lots are equipped with chargers.
- Scenario 2: Some parking lots are equipped with chargers.

Considering these two scenarios, we also test additional solution strategies:

- RL: Using the trained neural network by RL, we can select the most probable solution by the *greedy* decoding strategy as done in the previous section.
- Sample RL: We can generate solution samples from the trained neural network by the *probabilistic* decoding strategy instead of the greedy decoding. Among all solution samples, we choose the best solution.

- RL-ALNS: We can feed the RL solution by the greedy decoding strategy as the initial solution to ALNS.

3.7.2.1 Scenario 1

In this scenario, no parking lots are equipped with chargers. The demand, supply, and charge nodes are located at different places. The average objective values (makespan) and computational times on 143 days data are summarized in Table 3.5. For all instances, ALNS can perform better than EBNSM in both solution quality and computational times. ALNS can get less average objective values than RL for all instances. However, when each trained RL model is sampled 100 times in the Sample RL approach, a better solution can be obtained for instances (K, W) =(1,5), (1,7) and (1,9). We observe that RL-ALNS can obtain the best solution in all cases, while the computational times are decreased by up to 48.86%, compared to the Sample RL approach. Only in the instance (1,5), RL-ALNS consumes less computational time but produces a greater objective value.

Table 3.5: Average Makespan (Unit:Min) and Computational Times (Unit:Sec) in Scenario 1 for 143 Days Instances

P	(K, W)	EBNSM		RL		Sample RL		ALNS		RL-ALNS	
		obj	time	obj	time	obj	time	obj	time	obj	time
6	(1,5)	289.8	154.4	286.9	0.8	285.5	**79.8**	286.5	146.2	**280.1**	87.1
	(2,2)	287.5	152.3	270.2	0.9	269.7	91.2	267.1	142.9	**266.9**	**90.2**
	(3,1)	295.2	163.2	288.0	1.1	287.3	111.0	286.1	145.2	**284.0**	**91.2**
8	(1,7)	253.7	184.2	248.3	1.4	247.2	139.3	247.9	162.4	**245.9**	**96.6**
	(2,3)	234.5	152.3	228.7	1.0	226.4	101.2	224.7	98.4	**220.9**	**67.3**
	(4,1)	224.2	104.2	220.1	1.6	218.9	158.0	213.2	99.9	**212.1**	**63.0**
10	(1,9)	182.5	102.4	179.8	1.3	178.1	134.2	178.6	74.2	**177.2**	**67.2**
	(2,4)	218.7	110.4	205.8	1.2	203.4	119.8	201.3	82.1	**199.8**	**52.4**
	(5,1)	204.2	99.3	197.3	1.3	194.2	134.2	190.9	90.4	**187.3**	**66.6**

3.7.2.2 Scenario 2

In this scenario, many parking lots are equipped with chargers. Small changes are made to let ALNS adapt to Scenario 2. There are two cases:

- Many demand nodes are equipped with chargers.

  All EVs can directly move to one demand node that is equipped with a charger. The charge time is not necessary to be considered. The changes in ALNS only happen in finding EV relocation solution X as follows.

  – In Algorithm 2, remove Line 7-12 and only keep Line 11-12.

  – In the Greedy and Probabilistic s-d matching, $\mathcal{S}^c = \emptyset$.

- Many supply nodes are equipped with chargers.

  EVs at the supply nodes with chargers do not need to move to a charge station. EVs at these supply nodes are set into $\mathcal{S}^n$.

Table 3.6: Average Objective Values and Computational Times in Scenario 2

P	(K, W)	EBNSM		RL		Sample RL		ALNS		RL-ALNS	
		obj	time	obj	time	obj	time	obj	time	obj	time
6	(1,5)	223.7	145.8	214.3	1.3	213.8	131.3	202.5	122.8	**201.4**	**83.4**
	(2,2)	227.2	149.7	214.6	1.1	212.9	111.2	201.6	126.1	**200.9**	**78.2**
	(3,1)	200.3	134.2	199.5	1.4	198.4	138.3	199.4	156.2	**197.9**	**108.2**
8	(1,7)	182.1	155.1	153.6	1.5	151.8	152.3	145.2	133.7	**140.2**	**88.5**
	(2,3)	213.7	162.5	190.7	1.6	188.7	154.2	182.4	99.6	**180.3**	**76.9**
	(4,1)	200.5	118.5	188.3	2.0	187.0	198.4	187.4	100.1	**185.3**	**85.2**
10	(1,9)	125.1	111.2	104.3	1.9	102.7	187.3	99.3	77.8	**98.1**	**49.4**
	(2,4)	157.2	99.4	142.2	1.8	140.2	179.2	140.1	78.4	**138.7**	**55.9**
	(5,1)	129.5	99.4	120.2	2.1	118.2	209.3	111.4	89.7	**110.6**	**58.3**

Besides the changes stated above, the calculation of makespan also changes. When a shuttle arrives at a supply node equipped with a charger and drops off a worker, this worker

has to wait for EV to complete fully charging. So the EV's arrival time at supply $s \in \mathcal{S}^c$ becomes:

$$e_s \geq \tau_s + \frac{1}{\beta}\left(100 - l_s\right) \quad \forall s \in \mathcal{S}^c$$

The average objective values and computational times on Scenario 2 are summarized in Table 3.6. When the trained RL model is sampled 100 times, the average objective values of Sample RL become better than RL for all instances. When the RL solution is used as the initial solution in ALNS, the average objective values and computational times decrease by 1.08% and 30.52%.

3.7.3 Routing with Personal Mobility Vehicle

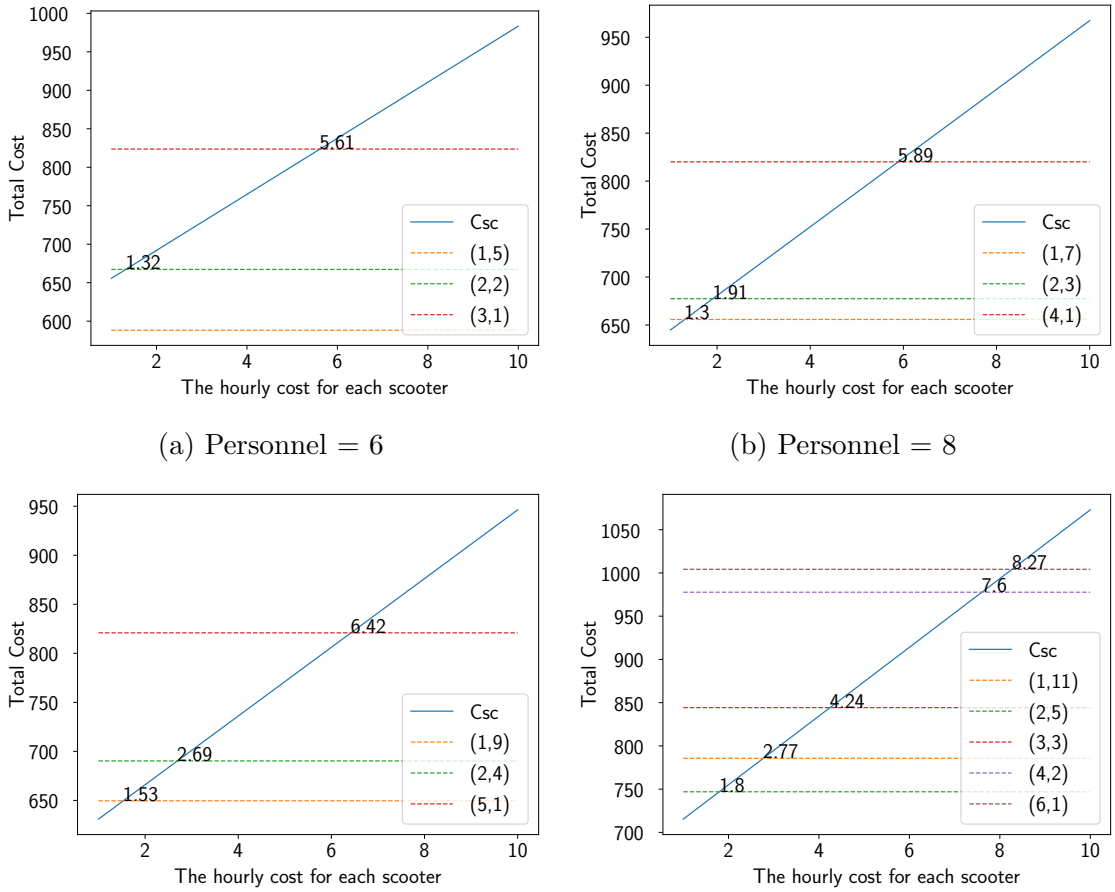
When a personal mobility vehicle is used instead of a shuttle, ALNS can solve the problem as well by following the changes in Section 3.6.1. The results are summarized in Table 3.7. The speed of scooters is assumed as 15mph. Since riding a scooter is much slower than a shuttle, the makespans for personnel 6, 8, 10 are all longer. The workers spend more time in the movement. Moreover, makespan is not the only factor that affects the decision. In this section, the analysis of total operation cost and the wait times are discussed between using shuttles or scooters.

Table 3.7: Average Objective Values and Computational Times of ALNS Using Scooters

P	Scenario1		Scenario2	
	obj	time	obj	time
6	364.2	100.2	303.2	98.6
8	268.7	120.4	200.2	113.2
10	210.3	140.5	183.4	145.2

3.7.3.1 Analysis on Total Operation Cost

Besides the total time spent in the system, the operation cost of EV relocation is also important in the FFEVSS. The cost consists of operating the fleet of shuttles or scooters and



(a) Personnel = 6

(b) Personnel = 8

(c) Personnel = 10

(d) Personnel = 12

Figure 3.5: Total Cost for Using Shuttles and Scooters

the labor cost of workers. Let T be the makespan value. Let Γ_w be the hourly cost for each worker. The median pay in 2020 for one vehicle driver was \$16.67 per hour (U.S. Bureau of Labor Statistics, 2021). Assume the per hour labor cost Γ_w is \$17/hr. Let Γ_{sh} and Γ_{sc} be the hourly cost to operate each shuttle and each scooter. The total cost when using shuttles is calculated as $C_{sh} = P \times T \times \Gamma_w + K \times T \times \Gamma_{sh}$. When using scooters, the total cost is calculated as $C_{sc} = P \times T \times (\Gamma_w + \Gamma_{sc})$. Assume per hour cost of a shuttle Γ_{sh} is \$24/hr.

The total cost for shuttles and scooters on Scenario1 is analyzed in the following. The total costs for the given number of personnel 6, 8, 10, 12 are illustrated in Figure 3.5. When Γ_{sc} is less than the hourly cost at the cross point, the total cost with a scooter is lower. It is better to choose the scooter as the movement tool.

The higher the cost Γ_{sc} at the cross point is, the better to choose scooters. It is better to choose scooters as the movement tool when the shuttle combination is (3,1), (4,1), and (5,1) for personnel 6, 8, and 10, respectively. For personnel 12, using scooters is better when the shuttle combinations are (6,1) and (4,2). Even though the more number of shuttles results in short makespan, the cost for shuttles becomes expensive. Given the certain number of personnel, more number of shuttles with small capacity is not a good choice. If the per hour labor cost increases, the makespan will also further impact the total cost. It is important to balance makespan and the cost of movement tools. This suggests not to use very large shuttles or scooters for the system with the high hourly labor cost.

3.7.3.2 Analysis on Wait Times

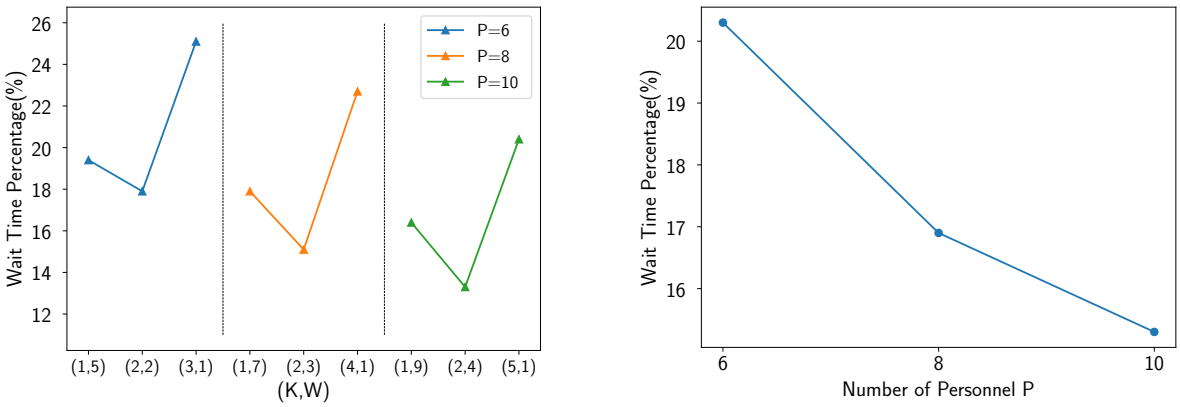
Makespan values are influenced by both wait times and movement times. When the mobility tool is the shuttle, the wait times happen at charge stations and demand nodes. If a shuttle arrives earlier than EV's arrival time, the shuttle has to wait for picking up the worker; otherwise, if EV arrives earlier than the shuttle's arrival time, the worker has to wait for a shuttle. So, the wait time per shuttle is calculated as $\frac{\sum_{i \in \mathcal{C} \cup \mathcal{C}^+ \cup \mathcal{D}} |\tau_i - e_i|}{K}$.

When the mobility tool is the scooter, workers can directly leave charge stations and demands by themselves. The wait times only happen at dummy charger nodes $i \in \mathcal{C}^+$. If a worker arrives earlier, he has to wait for the EV to complete charging; otherwise, the EV has to wait for a worker to drive it. The wait time per scooter is calculated as $\frac{\sum_{i \in \mathcal{C}^+} |\tau_i - e_i|}{P}$.

The wait time percentage is used to standardize the wait times as a percentage of the total time. The wait time percentage is calculated as

$$\text{Wait Time Percentage} = \frac{\text{Total Wait Time}}{\text{Makespan}} \times 100\%$$

The average wait time percentages using shuttles and scooters are illustrated in Figure 3.6. As shown in Figure 3.6a, given a certain number of personnel, when the number of



(a) Shuttles

(b) Scooters

Figure 3.6: Average Wait Time Percentage

shuttles is 2, the waiting time percentage is the lowest. When the number of shuttles is 1, the workers spend more time in waiting for the shuttle to pick them up. When the number of shuttles is more than 2, the shuttles have only one worker on board and have to pick up and drop off workers frequently. With the increasing number of personnel, the workers can do the relocation simultaneously, so the wait times also decrease. Thus, in order to decrease the wait time, it is important to choose the shuttles with proper capacity. As shown in Figure 3.6b, the wait time percentages become smaller with the increasing number of personnel. Because the scooter speed is slow, the workers spend more time on the movement, and EVs need to wait for a worker to drive it to the destination after charging. So, the more workers participate in the relocation assignment, the less wait time is.

3.7.4 EV Relocation and Routing in Dynamic Environment

This section compares the quality of the solution with the benchmark, assuming that the upcoming change is foreseen at the beginning of the planning. We test the dynamic environment when 50% EV relocations have been done. Two cases are considered: 1) Randomly remove 10% EV demands; 2) Randomly add 10% additional EV demands. The average makespan and computational times of 143 days Amsterdam data are summarized in the Column *obj* and *time* in Table 3.8. The objective values are calculated from the time when

Table 3.8: Results on Amsterdam Data in Dynamic Environment

P	(K, W)	Remove 10% EVs						Add 10% EVs					
		obj	time	5s	10s	20s	30s	obj	time	5s	10s	20s	30s
6	(1,5)	243.1	47.1	265.9	253.6	248.7	244.0	303.4	53.7	365.7	320.4	309.7	308.2
	(2,2)	238.5	46.3	265.7	243.9	240.5	238.5	298.6	56.0	332.5	313.8	303.7	299.8
	(3,1)	247.0	48.2	276.8	254.8	249.4	248.1	320.8	59.8	367.7	343.4	330.1	325.6
8	(1,7)	216.5	43.2	257.4	234.5	220.3	217.6	267.9	46.4	299.5	278.1	270.7	269.7
	(2,3)	199.5	44.5	243.5	210.2	201.3	199.8	254.8	49.5	276.7	261.7	258.7	255.2
	(4,1)	187.6	39.2	223.9	198.2	188.4	187.9	232.9	48.7	268.3	247.4	238.2	234.1
10	(1,9)	159.5	37.4	176.9	163.4	160.2	159.5	195.8	45.8	232.5	209.5	198.4	196.0
	(2,4)	176.3	29.7	199.6	180.5	177.5	176.3	220.8	36.8	249.7	232.2	222.4	220.8
	(5,1)	165.4	32.5	188.3	169.7	166.2	165.4	206.8	33.8	246.8	211.5	209.6	206.8
Avg. PD*				14.58	4.06	0.98	0.19			12.80	4.99	1.66	0.55

* Avg. PD is the Average Percentage Deviation between obj values and those under the limited time of 5s, 10s, 20s, 30s.

the shuttle departs the depot at the beginning. The computational times are the run times of getting a new solution when removing or adding EVs. When the stop criterion is set as a limited run time of 5s, 10s, 20s, and 30s, the average objective values are shown in Table 3.8. ALNS can solve the dynamic case when adding or removing EVs in the middle of EV relocation operations. ALNS can also provide a new routing for shuttles in short computational times, less than 48.2s for removing 10% EVs and less than 59.8s for adding 10% EVs. When 10% EV demands are removed, the average percentage deviations are 14.58%, 4.06%, 0.98%, 0.19% within the computational times 5s, 10s, 20s, and 30s, respectively. When 10% EV demands are added, average percentage deviations are 12.80%, 4.99%, 1.66%, and 0.55%. It shows that a relatively good new solution (less than 2%) can be obtained in a short time. Therefore, ALNS is a flexible method to adapt to the dynamic environment where some EVs are assigned to external drivers or additional EV demands are added in the middle of relocation operations.

3.8 Concluding Remarks

This chapter considers the EV relocation and shuttle routing problem for the rebalancing operation of free-floating EV sharing systems. One of the key operational decisions for the

carsharing company is how to relocate the EV fleet to meet the next day's demand with sufficient battery levels.

We develop a metaheuristic ALNS algorithm for the EV relocation problem that determines where to relocate each EV and how to route the shuttles that transport the staff drivers synchronously. We apply our method to conduct numerical experiments using both randomly generated data and actual FFEVSS data in Amsterdam. We found that ALNS outperforms EBNSM both in the solution quality and the computational time. Our ALNS also produces better solutions than the RL approach but requires much longer computational time than RL. Our experiments reveal that providing the RL solution as the initial solution for ALNS is an effective and efficient solution strategy that can take advantage of both approaches, achieving the best solution quality and reducing the computational time significantly.

We also demonstrate how our ALNS can be modified to solve the problem where staff drivers carry a personal mobility vehicle such as a scooter. Our further analysis provides practical recommendations on which mode of transportation will be more efficient—i.e., a small number of shuttles with large capacity or a large number of shuttles with small capacity (or even personal mobility)—in terms of total operational cost as well as wait times.

Lastly, we show that our ALNS that destroys an incumbent solution partially and repairs to a new solution in each iteration is quite flexible to be applied to a dynamic environment. Specifically, our numerical results highlight the usefulness of our flexible ALNS method for an environment where some EV demands are removed or added in the course of EV relocation operations.

As directions of future research, this model can be extended for day-time static relocation. Extending this model to the 24-hour period will ordinarily require redeployment of the model at constant, and relatively small, time intervals and also the assumption of zero new arriving demand. In that case, unlike our numerical experiments conducted with constant travel speed for the city of Amsterdam, a more robust analysis with different shuttle travel speeds can

be considered to account for various traffic conditions at different times and across different locations. An important factor in the successful implementation of static repositioning is the accuracy of the demand forecast. The demand faced by a car-sharing system is highly sensitive to a variety of external factors. In this study, we base our demand forecast on past demand data on similar days and focus on synchronous modeling of relocation and routing operations. However, more sophisticated data mining models and demand prediction models can be devised.

Chapter 4: An Adaptive Large Neighborhood Search Method for Drone-Truck Arc Routing Problem

4.1 Introduction

With the rapid development of unmanned aerial vehicles or drones, the use of advanced techniques increases the city level and improves the quality of life. Drones (or unmanned aerial vehicles) are recently widely applied in many fields such as aerial imaging (Rakha and Gorodetsky, 2018), traffic monitoring (Li et al., 2018), infrastructure inspections (Otto et al., 2018), policing and surveillance (Engberts and Gillissen, 2016), rescue (Rabta et al., 2018), product deliveries (Boysen et al., 2018; Wang and Sheu, 2019), and agriculture (Mogili and Deepak, 2018). The use of drones can improve service because of the higher speeds, lower cost, and safety. Because the drone can travel directly between any two nodes, it can fly along or off the roads. The drone is not limited to the ground transportation infrastructure while servicing the edges. The cooperation of the truck and the drone allows to adapt to the specific circumstances where some edges require service, but there are no roads, such as inspection along the power lines or pipelines (Yu et al., 2019). For example, some electric power lines in mountain areas are not accessible by ground vehicles, while other power lines have roads. The required edges can be covered by drones or trucks or both of them. Thus, the Drone-Truck Arc Routing Problem (DT-ARP) extends the traditional arc routing problem where the service is not limited to the road network and is done by the truck and the drone cooperatively.

In this chapter, DT-ARP problem optimizes the truck route and drone route to minimize the total time of completing all the tasks (all required edges are traversed at least once). Despite the benefits of DT-ARP, it is a complicated problem to make arc routing decisions.

The cooperation between the truck and the drone poses multiple challenges. First, the decisions on the truck's route and the drone's route are hierarchical and interdependent. The decision on the drone's takeoff and landing nodes depends on the truck route. Meanwhile, the truck must move along the route that includes takeoff and landing nodes. Second, because the drone has limited battery capacity, each flight trip has a physical constraint, i.e., maximum flight range. The drones must land on the truck frequently, and the driver replaces the battery for the drones. Third, it is allowed to fly over multiple arcs in one flight trip as long as the flight length is less than the maximum flight range. So, it is the uncertain number of arcs in each flight trip.

The Drone-Truck Arc Routing Problem in this paper is NP-hard because it is a special case of the Rural Postman Problem (RPP) that has been proved to be NP-hard by Lenstra and Kan (1976). To the best of our knowledge, there are few papers to look for the optimal solution for this kind of complicated NP-hard problem. Starting from a simple case, One-Drone-One-Truck is considered. Because we aim to minimize the total completion time (makespan), it is vital to obtain the arrival time at each node. However, it is hard to record the arrival time at each node, because each required edge is allowed to be traversed at least once, and thus the number of visits at each node is unknown. Therefore, in order to formulate a mathematical model for the One-Drone-One-Truck ARP, the arc routing problem is transformed into a standard vehicle (node) routing problem (VRP). Two kinds of transformation rules by Pearn et al. (1987) and Longo et al. (2006) are used here. Pearn et al. (1987) added two side nodes and one middle node over each required edge and ensured that each edge is traversed when all nodes are visited once. Longo et al. (2006) added two side nodes over each required edge and ensured that each edge is traversed when the side nodes are visited in sequence. The transformations are described in detail in Section 4.3.1. The optimal objective value in VRP is equivalent to that in ARP. The optimal ARP solution is obtained by being transformed back from the optimal VRP solution.

For a successful operation of DT-ARP, it is essential to develop an efficient method to solve the Drone-Truck arc routing problem, because DT-ARP is always large-scale in the real life. We develop an adaptive large neighborhood search (ALNS) algorithm to solve the problem. ALNS, first proposed by Ropke and Pisinger (2006), is a well-known iterative metaheuristic framework that has been popularly applied to solving various vehicle routing problems. ALNS was first applied to the arc routing problem by Laporte et al. (2010) who solved the capacitated arc routing problem with stochastic demands and multiple vehicles to minimize the total cost. The key characteristic of ALNS is to destroy an incumbent solution and repair it to construct a new solution in each iteration. The choices of destroy and repair method are determined adaptively by their previous successes. Appling ALNS on DT-ARP is not straightforward. Because two decisions of the truck route and drone route are tangled, ALNS should be modified to be able to handle such complexity.

Numerical experiments are conducted for which we use randomly generated instances of several sizes and a set of large-size benchmark undirected rural postman problem instances (Corberán et al., 2021). The performance of ALNS is shown by comparison with the optimal solution of the MIP formulation solved by Gurobi. Furthermore, ALNS solves a more complex case Multi-Drones and One-Truck over the randomly generated instances. A metaheuristic, named multi-start tabu search (MSTS) is proposed by Luo et al. (2021) to investigate the multi-visit traveling salesman problem with multi-drones and a truck with the aim to minimize makespan. With MSTS as being benchmark method, we demonstrate the effectiveness and efficiency of ALNS algorithm.

The remainder of the chapter is written as follows. In Section 4.2, the related literature is reviewed. The problem statement and mathematical model are presented in Section 4.3. Adaptive Large Neighborhood Search is described in detail in Section 4.4. In Section 4.5, the experimental results validate the performance of ALNS to solve the Drone-Truck Arc Routing Problem. Conclusions are summarized in Section 4.6.

4.2 Literature Review

There have been numerous studies to investigate optimization for the arc routing problem. Although there is a rapidly growing literature on the arc routing problems with trucks or drones, the research on the cooperation between the truck and the drone has been previously assessed only to a limited extent.

Some literature papers discussed the arc routing problem only with a single vehicle or a fleet of homogeneous vehicles. Hertz et al. (2000) proposed a tabu search heuristic for the capacitated arc routing problem where all given required edges were serviced exactly by one vehicle and minimize the total weight of all service edges. Tagmouti et al. (2010) proposed a variable neighborhood descent heuristic for a capacitated arc routing problem to minimize the time-dependent service costs. They transformed the problem into an equivalent vehicle routing problem and used it as an alternative approach. Then, Tagmouti et al. (2011) studied the same problem in the dynamic environment. After the vehicles started to work, new information showed up that affected costs. The variable neighborhood descent heuristic was adapted to this dynamic variant. Benavent et al. (2014) solved k-vehicles windy rural postman problem with minimizing the maximal distance traveled by a vehicle to find k-routes that service all the required edges in a windy graph. They proposed a branch-and-cut algorithm when the small number of vehicles and required edges. Vincent and Lin (2015) proposed an iterated greedy heuristic for the time-dependent prize-collecting arc routing problem and gave a vehicle route to maximize the profit. Monroy-Licht et al. (2017) proposed an adaptive large neighborhood search algorithm to solve the rural postman problem with time windows of serving some required edges with one vehicle and solved a set of large instances with up to 104 required edges. Calogiuri et al. (2019) proposed a branch and bound method to solve the time-dependent Rural postman problem in which the costs depend on the time.

There are a few papers considering the arc routing problem with drones and presenting an exact method for small scale and heuristic for large scale. Oh et al. (2011, 2014) modified a

road network search problem as a Multi-choice Multidimensional Knapsack problem to minimize flight time for multiple heterogeneous drones. And they proposed a greedy insertion metaheuristic method to produce the shortest path in consideration of physical constraints via the Dubins path planning. Dille and Singh (2013) also used Dubins path planning to optimize the drone routing where the drone has a sensor with a radius of coverage. The arc covering problem was converted into TSP by splitting the road network into a set of coverage points. The visits on these points ensure part of the road is covered within the range of the sensor. Chow (2016) formulated a deterministic arc-inventory routing problem for UAV-based traffic monitoring. And they also modeled the uncertain demand based on real-time data and derived a stochastic dynamic policy. An approximate dynamic programming algorithm based on the Least Squares Monte Carlo simulation was proposed and was validated better than the static myopic policy for small instances. Li et al. (2018) also explored an arc inventory routing and combined with capacitated arc routing with uncertain demand for traffic monitoring. The mixed-integer programming model was presented with the aim of minimizing the total cost. It solves up to 12 nodes and 40 lines with the Cplex solver. The real case study of road traffic in Shanghai is done by applying a local branching method. Campbell et al. (2018) studied drone arc routing problems to minimize the total cost where drones can travel directly between any two points and approximate each curve in the plane by a polygonal chain. The drones leave and enter at the points of the polygonal chain. An iterative algorithm was proposed to solve RPP instances with an increasing number of points of the polygonal chain. Campbell et al. (2021) also digitized the Length Constrained K-Drones Rural Postman Problem by a polygonal chain with a finite number of points. They presented a formulation and some valid inequalities. Based on this, they designed a branch-and-cut algorithm for small-size instances and a metaheuristic for large-size instances.

As stated above, a variety of methods have been proposed for solving the arc routing problem with the truck or the drones. Although there are an increasing number of papers

that address node routing problems with the truck and the drone (Boysen et al., 2018; Agatz et al., 2018; Khoufi et al., 2019; Wang and Sheu, 2019; Macrina et al., 2020; Chung et al., 2020; Leon-Blanco et al., 2022), the papers remain few that propose the exact or heuristic algorithm to solve the synchronization of drone and truck in arc routing problem.

4.3 Problem Statement

The Drone-Truck Arc Routing Problem can be described as follows. The notation is listed in Table 4.1. Let $G = (\mathcal{N}, \mathcal{E})$ be an undirected connected graph. The depot is labeled node 1. Define $\mathcal{R}$ as the set of required edges. One truck and one drone cooperatively traverse all required edges at least once. The aim is to find the truck and drone routes to minimize the makespan, i.e., the time leaving from and returning to the depot. There are some assumptions about the drone.

- Assumption 1. Because the drone can fly off the edge, the flight network is larger than the actual road network G. Define the set of drone flight edges $\mathcal{E}_d$ consists of all available paths between any two nodes $i, j \in \mathcal{N}$, in regards to the Drones Rules and Regulations. The drone follows the graph $G_d = (\mathcal{N}, \mathcal{E}_d)$. The distance of drone flight is calculated as the horizontal distance.

- Assumption 2. The drone has a maximum flight range because of the limited battery capacity. The drone must fly back to a truck before the battery runs out. After the drone lands at the truck, the driver replaces a full backup battery and makes sure the drone is prepared for the next trip.

- Assumption 3. The times for the drone to launch and land are neglected. The time to replace the battery is also neglected.

The formulation for the Drone-Truck arc routing problem is made in two stages: (1) Transform ARP into corresponding standard VRP; (2) Formulate mixed integer programming for the Drone-Truck Vehicle Routing Problem (DT-VRP). The objective value in DT-

VRP is equivalent to that in DT-ARP. The truck route and the drone route can be obtained by transforming back from VRP solution to ARP solution.

Table 4.1: Mathematical Notation

Sets	
G	Original undirected graph, $G = (\mathcal{N}, \mathcal{E})$
G_d	Original undirected flight network, $G_d = (\mathcal{N}, \mathcal{E}_d)$
$\mathcal{N}$	Set of original N vertices, $\mathcal{N} = \{1, 2, ..., N\}$
$\mathcal{E}$	Set of original undirected edges
$\mathcal{E}_d$	Set of original undirected edges for the drone flight
$\mathcal{R}$	Set of original undirected required edges $\mathcal{R} \subset \mathcal{E}$
H	The complete undirected graph of the corresponding VRP
$\mathcal{V}_H$	Set of the constructed VRP nodes which are transformed from ARP
$\mathcal{V}$	Set of VRP nodes, $\mathcal{V} = \mathcal{V}_H \cup \{N_v + 1\}$
$\mathcal{V}_1$	Set of VRP nodes excluding the depot, $\mathcal{V}_1 = \mathcal{V}_H \setminus \{1\}$
$\mathcal{A}$	Set of VRP directed arcs
$\mathcal{R}_H$	Set of VRP undirected required edges
Parameters	
v_t	Truck Speed
v_d	Done Speed
$\mathbf{d^T}$	The distance matrix between any two node in $\mathcal{V}_H$ for the truck
$\mathbf{d^D}$	The distance matrix between any two node in $\mathcal{V}_H$ for the drone
t_{ij}^T	The time of traversing arc $(i, j) \in \mathcal{A}$ for the truck
t_{ij}^D	The time of traversing arc $(i, j) \in \mathcal{A}$ for the drone
e	Maximum units of consecutive flight time, $e = \frac{\text{Maximum Drone flight range}}{v_d}$
Variables	
x_{ij}^T	1, if the truck traverses arc $(i, j) \in \mathcal{A}$; Otherwise, 0.
x_{ij}^D	1, if the drone traverses through arc $(i, j) \in \mathcal{A}$; Otherwise, 0.
y_i^T	1, if node $i \in \mathcal{V}_1$ is visited only by the truck; Otherwise, 0.
y_i^D	1, if node $i \in \mathcal{V}_1$ is visited only by the drone; Otherwise, 0.
y_i^C	1, if node $i \in \mathcal{V}_1$ is combined node where a drone launches or lands; Otherwise, 0.
$n_i^T, n_i^D \in \mathbb{Z}+$	The ordered visit sequence of nodes for the truck or the drone; Otherwise, 0.
$f_i \in \mathbb{R}+$	The flight time when the drone arrives at node $i \in \mathcal{V}$.
$a_i \in \mathbb{R}+$	The arrival time of the truck or the drone at node $i \in \mathcal{V}$.

4.3.1 Transformation ARP to VRP

We apply two kinds of the arc-to-node transformation proposed by Pearn et al. (1987) and Longo et al. (2006). Pearn et al. (1987) replaced each required edge with three vertices to transform ARP into the corresponding VRP. Longo et al. (2006) eliminated one of every three nodes and achieved the same objective with specific constraints (each required edge must be traversed at least once). The details are shown in the following section.

4.3.1.1 Pearn et al. (1987) Transformation

Pearn et al. (1987) transformed ARP into VRP by replacing each required edges $(i, j) \in \mathcal{R}$ by two side nodes s_{ij}, s_{ji} and one middle central node m_{ij}. The corresponding VRP is defined on the complete undirected graph $H = (\mathcal{V}_H, \mathcal{E}_H)$.

$$\mathcal{V}_H = \bigcup_{(i,j)\in\mathcal{R}} \{s_{ij}, m_{ij}, s_{ji}\} \cup \{1\}$$

$$\mathcal{E}_H = \{(i, j) : i \neq j, i, j \in \mathcal{V}_H\}$$

The set of VRP nodes includes every three nodes of each required edge and the depot node 1. The ARP with $|\mathcal{R}|$ required edges is transformed into the undirected complete graph VRP with $3 \times |\mathcal{R}| + 1$ nodes.

The distance of the edges in H are defined as the below equations.

$$d(s_{ij}, s_{kl}) = \begin{cases} 0 & \text{if } (i, j) = (k, l) \\ c(i, j) & \text{if } (i, j) = (l, k) \\ \text{dist}(i, k) & \text{if } (i, j) \neq (k, l), (i, j) \neq (l, k) \end{cases}$$

$$d(1, s_{ij}) = \text{dist}(1, i)$$

$$d(m_{ij}, v) = \begin{cases} \frac{1}{2}c_{ij} & \text{if } v = s_{ij} \text{ or } s_{ji} \\ \infty & \text{otherwise} \end{cases}$$

where $\mathsf{dist}(i, j)$ is the shortest path distance between node i and j in the original graph. The purpose of the middle node m_{ij} is to ensure that the shortest path between two side node s_{ij} and s_{ji} is always $s_{ij} \to m_{ij} \to s_{ji}$ or $s_{ji} \to m_{ij} \to s_{ij}$ in sequence.

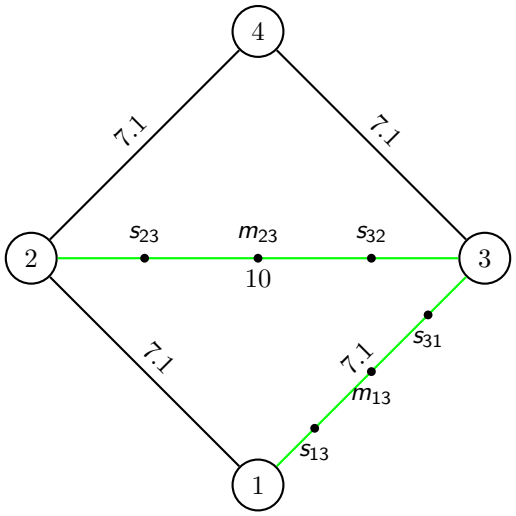


Figure 4.1: An Example for Arc Routing Problem with New Nodes. (Original arc distance are shown next to each arc; green lines mean the required edges)

It is noted that the distance calculation rule is different from that in Pearn et al. (1987)'s paper. The objective function in this paper is to minimize the completion time, while the objective in Pearn et al. (1987)'s paper is to minimize the total cost. The arrival time at each node is requested to be accurate. Although the three nodes are placed over required edges at different locations (shown in Figure 4.1), they are actually dummy and the distances are calculated based on their real locations. For example, the set of side nodes $\{s_{ij}, \forall i \in G : (i, j) \in \mathcal{R}\}$ are the dummy nodes for the real node $i \in G$. Their real locations are actually at the real node i. In the example Figure 4.1, s_{32} and s_{31} are the dummy nodes to node 3. Their real locations are set at node 3.

4.3.1.2 Longo et al. (2006) Transformation

Longo et al. (2006) replaced each required edge with two vertices. An edge $(i, j) \in \mathcal{R}$ is associated to vertices s_{ij} and s_{ji}. The pass through edge (i, j) is same as visiting two vertices s_{ij} and s_{ji} in sequence ($s_{ij} \rightarrow s_{ji}$ or $s_{ji} \rightarrow s_{ij}$). The ARP problem can be solved just by working out the corresponding $2 \times |\mathcal{R}| + 1$ VRP problem where each node is visited exactly once. The corresponding VRP is defined on the complete undirected graph $H = (\mathcal{V}_H, \mathcal{E}_H)$.

$$\mathcal{V}_H = \bigcup_{(i,j)\in\mathcal{R}} \{s_{ij}, s_{ji}\} \cup \{1\}$$

The distances between internodes are calculated as the following equations (Longo et al., 2006).

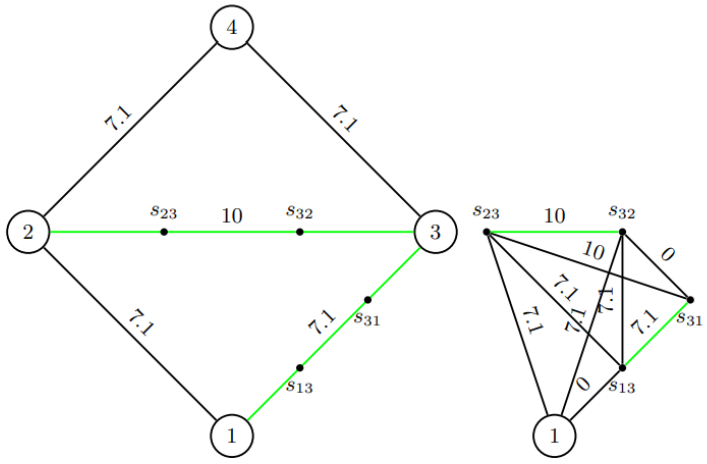
$$d(s_{ij}, s_{kl}) = \begin{cases} 0 & \text{if } (i, j) = (k, l) \\ c(i, j) & \text{if } (i, j) = (l, k) \\ \mathsf{dist}(i, k) & \text{if } (i, j) \neq (k, l), (i, j) \neq (l, k) \end{cases}$$

$$d(1, s_{ij}) = \mathsf{dist}(1, i)$$

where $\mathsf{dist}(i, j)$ is the shortest path distance between node i and j. Since the network of the truck $G = (\mathcal{N}, \mathcal{E})$ and the network of the drone $G_d = (\mathcal{N}, \mathcal{E}_d)$ are different, the distance matrix for the truck $\mathbf{d}^{\mathsf{T}}$ and for the drone $\mathbf{d}^{\mathsf{D}}$ are calculated with regard to G and G_d, respectively.

The requirement of traversing the required edges can be satisfied by adding a specific constraint that is to visit s_{ij} and s_{ji} in sequence. So, in the graph H, define the set of the undirected required edges $\mathcal{R}_H$ as follows.

$$\mathcal{R}_H = \{(s_{ij}, s_{ji}) | (i, j) \in \mathcal{R}\}$$



(a) Original ARP graph G (b) Corresponding VRP complete graph H

Figure 4.2: Transformation Arc Routing Problem to Node Routing Problem. (Green lines mean the required edges; the values above edges are the distances between internodes)

4.3.2 MIP Formulation for Drone-Truck VRP

The MIP formulation for DT-VRP is built based on the compact formulation by Roberti and Ruthmair (2021). The main differences between my formulation and theirs are that (i) They set the arrival time at one node based on the assumption that the drone's speed is greater than the truck's speed. We do not need this assumption. We modify the arrival time restriction to Constraint (4.19), such that the time through one arc depends on the truck's speed when the drone gets aboard the truck. (ii) Their subtour elimination constraints (arrival times restriction) become invalid here, because some distances between internodes are zero in our problem, such as $d_{1,s_{ij}} = d_{s_{31},s_{32}} = 0$ shown in Figure (4.2). So, we introduce the variables n^T and n^D to denote the ordered sequence of visiting nodes. Constraints (4.15) and (4.16) are added to avoid causing subtours. (iii) They restricted that the drone can only visit one node in a single flight trip. Our formulation allows the drone to visit multiple nodes in a single trip.

The vertex set $\mathcal{V}$ is defined as $\mathcal{V} = \mathcal{V}_H \cup \{N+1\}$, where node $N+1$ represents the enter depot node. Define $\mathcal{V}_1 = \mathcal{V}_H \setminus \{1\}$ as the set of nodes excluding the depot. The undirected

edges are extended as directed edges $\mathcal{A} = \{(i,j)|i,j \in \mathcal{V}_H : i \neq j\} \cup \{(i,N+1)|i \in \mathcal{V}_H : i \neq 1\}$. The mathematical formulation is described as the following.

The Objective Function is to find routes of a truck and a drone to minimize the total completion time, i.e., makespan. The value of makespan is calculated between the times when the truck leaves and returns to the depot.

$$\text{minimize} \quad a_{N+1} \tag{4.1}$$

Required Edges must be traversed at least once by the truck or the drone (Constraint (4.2)). It is noted that Constraint (4.2) is only needed for the 2-node transformed VRP (Longo et al., 2006). This ensures that all originally required edges are traversed when all side vertices are visited. Constraint (4.3) restricts that truck routing decision x^T and drone routing decision x^D are binary variables.

$$x_{ij}^T + x_{ji}^T + x_{ij}^D + x_{ji}^D \geq 1 \qquad \forall (i,j) \in \mathcal{R}_H \tag{4.2}$$

$$x_{ij}^T, x_{ij}^D \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A} \tag{4.3}$$

Flows for Truck and Drone Routes are described as follows. The truck leaves and returns to the depot exactly once. Outflow at the depot leave node 1 is 1 and inflow at depot enter node $N+1$ is also 1 (Constraint (4.4)). Constraint (4.5) restricts the flow balance for the other nodes. Constraints (4.6) and (4.7) restrict the flow balance for the drone route.

$$\sum_{(1,j)\in\mathcal{A}} x_{1j}^T = \sum_{(i,N+1)\in\mathcal{A}} x_{i,N+1}^T = 1 \tag{4.4}$$

$$\sum_{(i,j)\in\mathcal{A}} x_{ij}^T - \sum_{(j,i)\in\mathcal{A}} x_{ji}^T = 0 \qquad \forall i \in \mathcal{V}_1 \tag{4.5}$$

$$\sum_{(1,j)\in\mathcal{A}} x_{1j}^D = \sum_{(i,N+1)\in\mathcal{A}} x_{i,N+1}^D = 1 \tag{4.6}$$

$$\sum_{(i,j)\in\mathcal{A}} x_{ij}^{D} - \sum_{(j,i)\in\mathcal{A}} x_{ji}^{D} = 0 \qquad \forall i \in \mathcal{V}_1 \tag{4.7}$$

Node Category is introduced as follows. Let y_i^T be a binary variable equal to 1 if node $i \in \mathcal{V}_1$ is only visited by the truck, called a truck node. y_i^D is a binary variable that is equal to 1 if node $i \in \mathcal{V}_1$ is only visited by the drone, called drone node. Let $y_i^C \in \{0,1\}$ be equal to 1 if node $i \in \mathcal{V}_1$ is visited by both the truck and the drone, called the combined node. Constraint (4.8) ensures that each node must be one of three categories of nodes. Constraints (4.9) and (4.10) ensure the drone takes off and lands at the combined nodes and allows the drone to visit multiple nodes in a single flight trip. Variable α_{ij} decides that arc (i,j) can form the drone routes in two cases: one endpoint is (i) combined node or (ii) drone node. In example Figure 4.3, arc (1,2) suits the first case: one endpoint is combined node; arc (2,3) suits the second case: one endpoint is drone node.

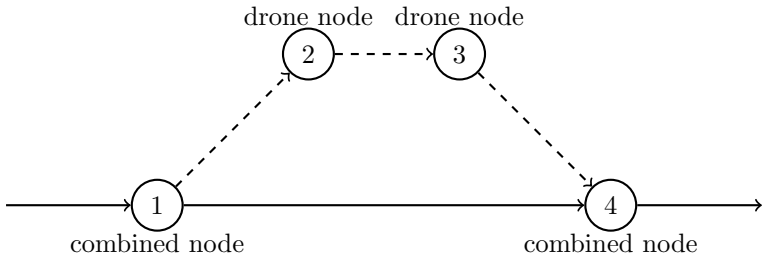


Figure 4.3: An Example for Node Category in One Flight Trip.
(Solid lines are truck route and dashed lines are drone route)

Constraints (4.11) and (4.12) link x_{ij}^T and x_{ij}^D with y and ensure that along the truck (drone) route, the node is either truck (drone) node or combined node. Constraints (4.13) and (4.14) ensure variable y and α are binary.

$$y_i^T + y_i^D + y_i^C = 1 \qquad \forall i \in \mathcal{V}_1 \tag{4.8}$$

$$x_{ij}^D + x_{ji}^D \leq y_i^C + y_j^C + 2(1-\alpha_{ij}) \qquad \forall (i,j) \in \mathcal{A} : i,j \notin \{1, N+1\} \tag{4.9}$$

$$x_{ij}^D + x_{ji}^D \leq y_i^D + y_j^D + 2\alpha_{ij} \qquad \forall (i,j) \in \mathcal{A} : i,j \notin \{1, N+1\} \tag{4.10}$$

$$\sum_{(i,j)\in\mathcal{A}} x_{ij}^T = y_i^T + y_i^C \qquad \forall i \in \mathcal{V}_1 \tag{4.11}$$

$$\sum_{(i,j)\in\mathcal{A}} x_{ij}^D = y_i^D + y_i^C \qquad \forall i \in \mathcal{V}_1 \tag{4.12}$$

$$y_i^T, y_i^D, y_i^C \in \{0,1\} \qquad \forall i \in \mathcal{V}_1 \tag{4.13}$$

$$\alpha_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A} \tag{4.14}$$

Subtour Elimination constraints are added to avoid causing subtours for the truck (Constraint (4.15)) and for the drone (Constraint (4.16)). Let n_i^T and n_i^D denote the sequence order of visiting nodes for the truck and the drone. Let $N_v = |\mathcal{V}_H|$.

$$n_j^T \geq n_i^T + N_v x_{ij}^T - (N_v - 1) \qquad \forall (i,j) \in \mathcal{A} \tag{4.15}$$

$$n_j^D \geq n_i^D + N_v x_{ij}^D - (N_v - 1) \qquad \forall (i,j) \in \mathcal{A} \tag{4.16}$$

$$n_i^T, n_i^D \in \mathbb{Z}+ \qquad \forall i \in \mathcal{V}_1 \tag{4.17}$$

Arrival Time of the truck or the drone is denoted as $a_i \in \mathbb{R}+$ at node $i \in \mathcal{V}$. Define t_{ij}^T and t_{ij}^D be the time of traversing arc (i,j) for the truck and the drone, respectively.

$$t_{ij}^T = \frac{d_{ij}^T}{v_t} \quad t_{ij}^D = \frac{d_{ij}^D}{v_d}$$

Constraints (4.18) and (4.19) set the arrival times of the truck and the done at the node. It is noted that when the drone gets aboard the truck ($x_{ij}^D = 1, x_{ij}^T = 1$), the arrival time at node j only depends on the truck's traverse time. Constraints (4.20) and (4.21) show that the total completion time cannot be lower than the summation of the traverse time by the truck or by the drone. Constraint (4.22) restricts that the arrival time is nonnegative continuous variable.

$$a_j \geq a_i + t_{ij}^T - M(1 - x_{ij}^T) \qquad \forall (i,j) \in \mathcal{A} \tag{4.18}$$

$$a_j \geq a_i + t_{ij}^D - M(1 - x_{ij}^D) - Mx_{ij}^T \qquad \forall (i,j) \in \mathcal{A} \tag{4.19}$$

$$\sum_{(i,j)\in\mathcal{A}} t_{ij}^T x_{ij}^T \leq a_{N+1} \tag{4.20}$$

$$\sum_{(i,j)\in\mathcal{A}} t_{ij}^D x_{ij}^D \leq a_{N+1} \tag{4.21}$$

$$a_i \geq 0 \qquad \forall i \in \mathcal{V} \tag{4.22}$$

Drone Flight Range is considered because of the limited battery. Let e be the maximum consecutive flight time. Constraint (4.23) guarantees that the drone can not traverse an arc whose flight time exceeds e unless the drone gets aboard the truck. A variable f_i is introduced to track the flight time in a flight trip. Constraint (4.24) sets the tracking flight time f_i. The flight time of a trip must be not greater than e (Constraint (4.25)).

$$x_{ij}^D \leq x_{ij}^T \qquad \forall (i,j) \in \mathcal{A} : t_{ij}^D > e \tag{4.23}$$

$$f_j \geq f_i + t_{ij}^D - M(1 - x_{ij}^D) - Mx_{ij}^T \qquad \forall (i,j) \in \mathcal{A} \tag{4.24}$$

$$0 \leq f_i \leq e \qquad \forall i \in \mathcal{V} \tag{4.25}$$

4.4 Adaptive Large Neighborhood Search

In this section, we develop an Adaptive Large Neighborhood Search (ALNS) for the Drone-Truck Arc Routing Problem. ALNS was first proposed by Ropke and Pisinger (2006) and applied to the vehicle routing problem - the pickup and delivery problem with time windows. ALNS is a well-known popular iterative algorithm to solve various vehicle routing problems. The idea of ALNS is to search in a neighborhood by destroying an incumbent solution and repairing it to construct a new solution in each iteration. The adaptivity is achieved by determining the choices of several destroy and repair methods on their previous successes. Laporte et al. (2010) first applied ALNS to solve the arc routing problem to minimize the total cost. To the best of our knowledge, this is the first paper using ALNS

on the Drone-Truck Arc Routing Problem to minimize the completion time that considers truck routing and drone routing jointly.

The procedure of the proposed ALNS is shown in Algorithm 4. DM and RM are denoted as the sets of the destroy and repair methods, respectively. The solution has two decisions: the truck route and the drone route. The key part is how to determine the sequence of traversing the required edges. Thus, it is vital to create X_r and Y_r which represent the sequence of required arcs traversed by the truck and the drone. In each iteration, the new X_r and Y_r in the neighborhood are produced by applying destroy and repair. The destroy process is to remove some edges from the truck required edges route X_r and drone required edges route Y_r. Next, the repair process can reconstruct the partial X_r and Y_r. Then, the complete truck route X and drone route Y are obtained by encoding from X_r and Y_r, described in Section 4.4.1. The destroy and repair methods are chosen by using the roulette-wheel selection principle based on their probabilities. When the method creates a better solution, the probability of the corresponding method increases, as described in section 4.4.5. The acceptance rule is used: the new best solution is accepted if its objective value is better than that of the current best solution; the new solution with a higher objective value has a chance to be accepted by the simulated annealing acceptance criterion. T denotes the value of the temperature and gradually decreases at each iteration by a rate $h \in [0, 1]$. The stop criteria are the maximum iterations $N_{\max}$ and non-improving iteration $Z_{\max}$.

4.4.1 Decoding and Encoding

The feasible solution in ALNS has two decisions: the truck route X and the drone route Y. The route is decoded by a string of arcs that represents the sequence of traversing the arcs. Let X_r and Y_r denote the sequence of required edges traversed by the truck and the drone, respectively. It is noted that the edges in X_r and Y_r do not have a direction.

Algorithm 4: Pseudocode for ALNS

Input: G, $\mathcal{R}$, DM, RM, $N_{\max}$, $Z_{\max}$
Output: X_{best}, Y_{best}

Initialize the truck required edges route X_{r0} and the drone required edges routes Y_{r0} (Sec 4.4.2);
Initialize destroy methods probability $\mathbf{P}_D^0$ and repair methods probability $\mathbf{P}_R^0$ (Sec 4.4.5);
$X_{r\text{best}} \leftarrow X_{r\text{current}} \leftarrow X_{r0}$, $Y_{r\text{best}} \leftarrow Y_{r\text{current}} \leftarrow Y_{r0}$;
Encode the required edges route into the complete route $X_{\text{best}}, Y_{\text{best}} \leftarrow \mathsf{encode}(X_r, Y_r)$ (Sec 4.4.1);
Calculate the makespan of current best solution $t_{\text{best}} \leftarrow f(X_{\text{best}}, Y_{\text{best}})$;
$N \leftarrow 1, Z \leftarrow 0$;
while $N \leq N_{\max}$, $Z \leq Z_{\max}$ **do**
    Select a destroy method $d \in \mathsf{DM}$ with probability P_D^N;
    Select a repair method $r \in \mathsf{RM}$ with probability P_R^N;
    Let X_{rnew} and Y_{rnew} be the new required edges solution obtained by appling destroy d and repair r on $X_{\text{rcurrent}}, Y_{\text{rcurrent}}$;
    Obtain the complete truck route and drone route $X_{\text{new}}, Y_{\text{new}} \leftarrow \mathsf{encode}(X_{\text{rnew}}, Y_{\text{rnew}})$;
    if $f(X_{\mathsf{new}}, Y_{\mathsf{new}}) < t_{\mathsf{best}}$ **then**
        $X_{\text{best}} \leftarrow X_{\text{new}}$, $Y_{\text{best}} \leftarrow Y_{\text{new}}$, $t_{\text{best}} \leftarrow f(X_{\text{new}}, Y_{\text{new}})$, $X_{\text{rbest}} \leftarrow X_{\text{rnew}}$, $Y_{\text{rbest}} \leftarrow Y_{\text{rnew}}$, $Z \leftarrow 0$;
    else
        $Z \leftarrow Z + 1$;
        $v = e^{-(f(X_{\mathsf{new}}, Y_{\mathsf{new}}) - f(X_{\mathsf{current}}, Y_{\mathsf{current}}))/T}$;
        Generate a random number $\epsilon \in [0, 1]$;
        if $\epsilon < v$ **then**
            $X_{\text{rcurrent}} \leftarrow X_{\text{rnew}}$, $Y_{\text{rcurrent}} \leftarrow Y_{\text{rnew}}$;
    $T \leftarrow hT$;
    Update $\mathbf{P}_D^N$ and $\mathbf{P}_R^N$ (Sec 4.4.5);
    $N \leftarrow N + 1$;

The encoding rule turns X_r and Y_r into the complete route solution X and Y. The encoding rule is done in two steps: (1) construct the complete truck route X; (2) connect the edges in Y_r to the truck route X.

In step 1, any two required edges in X_r are connected with the shortest path. Let $X_r = \{(v_a, v_{a+1}), (v_b.v_{b+1}), ...\}$. Starting from the depot 1, calculate the shortest distance $\mathsf{dist}(1, v_a)$ and $\mathsf{dist}(1, v_{a+1})$ and choose the shortest path to connect. If $\mathsf{dist}(1, v_a) < \mathsf{dist}(1, v_{a+1})$, add

$\{(1, v_a), (v_a, v_{a+1})\}$ in X; otherwise, add $\{(1, v_{a+1}), (v_{a+1}, v_a)\}$. If v is the last node in the current partial X, choose the shortest path $v \to v_b$ or $v \to v_{b+1}$ to append to the end in X. Complete the truck route until all required edges in X_r are done.

In step 2, Connect drone required edges Y_r to the truck route X by a greedy rule. Let $Y_r = \{(v_c, v_{c+1}), (v_d, v_{d+1}), ...\}$. First, extract all node from the truck route X as $V_x = \{1, v_1, v_2, ...\}$. Then, insert any edge in all possible locations between two consecutive nodes $\forall v_i, v_{i+1} \in V_x$. Choose the direction of drone edges with the smaller distance between $\mathsf{dist}(v_i, v_c) + \mathsf{dist}(v_{c+1}, v_{i+1})$ and $\mathsf{dist}(v_i, v_{c+1}) + \mathsf{dist}(v_c, v_{i+1})$. Calculate the increased value in the objective value between before and after inserting the edge. Choose the location with the least increased objective value to insert the edge (v_c, v_{c+1}). The partial Y becomes $\{.., (v_{i-1}, v_i), (v_c, v_{c+1}), (v_{i+1}, v_{i+2}), ...\}$. Next, put the vertices v_c, v_{c+1} in V_x. Repeat the above procedures until all required edges are done.

Two constraints must be satisfied to construct the drone route:

- The drone required edges can not insert between two nodes belonging to an arc in Y_r. For example, (v_d, v_{d+1}) can not insert between v_c and v_{c+1}.
- The flight distance must less than or equal to the maximum drone flight range.

4.4.2 Initial Solution

The initial X_r and Y_r are created by the Nearest Neighborhood Search. First, the required edges are randomly assigned to the truck set $\mathcal{S}_{xr}$ and the drone set $\mathcal{S}_{yr}$. Next, the required edges route is constructed progressively by adding the nearest edge. The procedure is shown in Algorithm 5. The distance between any two edges (a, b) and (c, d) shows the spatial closeness and is defined as following equation.

$$\mathsf{dist_e}\big((a, b), (c, d)\big) = \frac{1}{4}\big(\mathsf{dist}(a, c) + \mathsf{dist}(b, d) + \mathsf{dist}(a, d) + \mathsf{dist}(b, c)\big) \tag{4.26}$$

where $\mathsf{dist}(i, j)$ is the shortest path distance between node i and j in the graph G.

Algorithm 5: Pseudocode for initialization required edges route

Input: The set of the required edges for the truck or the drone $\mathcal{S}_r$
Output: The required edges route r

$r \leftarrow \{(a, b)\}$ with $(a, b) = \arg\min\{\text{dist}_e\big((1, 1), (a, b)\big), \forall (a, b) \in \mathcal{S}_r\}$;
while $\mathcal{S}_r \neq \emptyset$ **do**
    $(c, d) \leftarrow$ the last edge in r;
    Add the nearsest edge (u, v) to the end of r with
     $(u, v) = \arg\min\{\text{dist}_e\big((c, d), (u, v)\big), \forall (u, v) \in \mathcal{S}_r\}$;
    $\mathcal{S}_r \leftarrow \mathcal{S}_r \setminus (u, v)$;

4.4.3 Destroy Methods

Three destroy methods are applied to destroy a feasible solution. The destroy methods are Random Removal, Worst Route Removal, and Cluster Removal.

4.4.3.1 Random Removal

Randomly remove a certain percentage $q\%$ of edges from the truck and the drone required edges route X_r and Y_r.

4.4.3.2 Worst Route Removal

Given a solution (X_r, Y_r), the cost for the required edge e is defined as the difference value in the objective function before and after removing edge e from current solution. It is expressed as

$$\text{cost}(e, X_r, Y_r) = \max\Big\{f\big(\text{encode}(X_r, Y_r)\big) - f_{-e}\big(\text{encode}(X_r, Y_r)\big), 0\Big\} \tag{4.27}$$

where $f_{-e}(\cdot)$ is objective function after removing edge e after current solution. Sort all costs for required edge $e \in \mathcal{R}$ in the descending order. Remove the first $\lfloor q\% \times |\mathcal{R}| \rfloor$ edges with the larger costs from X_r and Y_r.

4.4.3.3 Cluster Removal

The idea of cluster removal is to avoid generating a similar new solution and try to jump into a farther neighborhood to get a solution with the large change. The relatedness between two edges (u, v) and (i, j) is measured by considering two factors: "distance" and "time". The "distance" represents the spatial closeness of these two edges and is calculated as Equation (4.26).

The "time" between two edges (u, v) and (i, j) represents temporal closeness and is defined as the average arrival times at start points and end points.

$$t((u, v), (i, j)) = \frac{1}{4}(|a_u - a_i| + |a_v - a_j| + |a_u - a_j| + |a_v - a_i|)$$

For any edge $(u, v) \in \mathcal{R}$, the measure of relatedness is defined as the following equations.

$$\begin{aligned} R((u, v), (i, j)) = & w_1 \frac{\mathsf{dist}_e((u, v), (i, j))}{\max\{\mathsf{dist}_e((u, v), (k, l)), \forall (k, l) \in \mathcal{R}\}} + \\ & w_2 \frac{t((u, v), (i, j))}{\max\{t((u, v), (k, l)), \forall (k, l) \in \mathcal{R}\} - \min\{t((u, v), (k, l)), \forall (k, l) \in \mathcal{R}\}} \end{aligned} \tag{4.28}$$

where w_1 and w_2 are weights with sum of 1.

The smaller $R((u, v), (i, j))$ is, the more related two edges are. Following steps are followed: randomly select a required edge $(u, v) \in \mathcal{R}$ and calculate $R((u, v), (i, j)), \forall (i, j) \neq (u, v) \in \mathcal{R}$. Sort all $R((u, v), (i, j))$ in descending order. Remove edges with first $\lfloor q\% \times |\mathcal{R}| \rfloor$ in the sequence.

4.4.4 Repair Methods

There are two repair methods to reconstruct the partial truck and drone required edges route X_r and Y_r. The repair methods are Random Insertion, Greedy Insertion, and Regret Insertion.

4.4.4.1 Random Insertion

Given partial truck required edges route X_r and drone required edges route Y_r, randomly insert the undecided required edges into them.

4.4.4.2 Greedy Insertion

The idea of the greedy insertion heuristic is to find the best insertion. A concept of Insertion Cost $I(e, p, X_r, Y_r)$ is introduced to denote the change in the objective value when inserting the edge e into Xr or Y_r at position p. It is expressed as

$$I(e, p, X_r, Y_r) = \Delta f(e, p, X_r, Y_r) \tag{4.29}$$

Select the position p to insert e with the least insertion cost $I(s, p, X_r, Y_r)$. Repeat the above steps until all required edges are inserted.

4.4.4.3 Regret Insertion

The regret insertion is improved by incorporating look-ahead information when selecting the required edge to insert.

For any required edge $e \in \mathcal{R}$, the regret-k cost is defined as

$$R(e, X_r, Y_r) = \sum_{j=1}^{k} \{\Delta f_j(e, X_r, Y_r) - \Delta f_1(e, X_r, Y_r)\} \tag{4.30}$$

where $\Delta f(e, X_r, Y_r)$ is the increased value in the objective value after inserting edge e. Sort $\Delta f(e, X_r, Y_r)$ for all possible insertion positions in the increasing order. $\Delta f_k(e, X_r, Y_r)$ means the increased value in the objective for the k-th best insertion position. The best insertion position has the least $\Delta f_1(e, X_r, Y_r)$. The regret insertion is the reconstruction heuristic that chooses to insert the required edge with the maximum $R(e, X_r, Y_r)$ and insert this edge into

the position with the least insertion cost. Repeat the above procedures until all required edges are inserted.

4.4.5 Adaptive Probability Update

The adaptivity of ALNS is achieved by selecting the destroy and repair methods based on their previous successes. In each iteration, the methods are chosen by the roulette wheel selection principle based on their probabilities.

The weights are introduced to track the scores to measure how well the methods have performed. The initial weights are equal to 1. In iteration i, the destroy method d and the repair method r are selected. If the methods creates a new global best solution, the weight $w_d^{i+1} \leftarrow w_d^i + \sigma_1$ and $w_r^{i+1} \leftarrow w_r^i + \sigma_1$; if the new solution is accepted with a better objective value than the current solution but not the global best one, $w_d^{i+1} \leftarrow w_d^i + \sigma_2$ and $w_r^{i+1} \leftarrow w_r^i + \sigma_2$; if the new solution is accepted with a worse objective value than the current solution, $w_d^{i+1} \leftarrow w_d^i + \sigma_3$ and $w_r^{i+1} \leftarrow w_r^i + \sigma_3$.

Then, the probability values of destroy and repair methods in iteration i are updated as follows:

$$\mathbf{P}_D^i = \left(\frac{w_d^i}{\sum_{d \in \mathsf{DM}} w_d^i} : d \in \mathsf{DM} \right) \tag{4.31}$$

$$\mathbf{P}_R^i = \left(\frac{w_r^i}{\sum_{r \in \mathsf{RM}} w_r^i} : r \in \mathsf{RM} \right) \tag{4.32}$$

4.5 Numerical Experiments

The experiments are implemented on the computer which has a 2.2GHz Intel Xeon Processor and 32GB RAM. The MIP formulation in Section 4.3.2 is solved in Julia v1.6.1 by calling Gurobi v0.9.12. Adaptive Large Neighborhood Search and Tabu Search are coded in Julia v1.6.1. The experiments are implemented on small-size randomly generated data and a set of large-size undirected rural postman problem instances.

4.5.1 One Truck and One Drone

4.5.1.1 Analysis on the Small-Size Instances

The data are randomly generated when the number of nodes $|\mathcal{N}| = 10, 15$, the number of edges $|\mathcal{E}| = 20, 30$, the number of required edges $|\mathcal{R}| = 5, 7, 10$. The vertices are randomly distributed in a 100×100 square region. The required edges are randomly chosen from all edges. Each type of randomly generated data has 25 instances. Define the maximum flight time $e = \beta \times \frac{1}{|\mathcal{E}_d|} \sum_{(i,j)\in\mathcal{E}_d} d_{ij}^D \div v_d$. The parameter β is set as 1,2,3 which determines the flight range. The speed of the truck v_t and the speed of the drone v_d are selected as equal (1,1), slower (1,2), and faster (2,1).

We use two kinds of transformation rule to convert ARP into $2 \times |\mathcal{R}| + 1$ and $3 \times |\mathcal{R}| + 1$ VRP. The One-Drone-One-Truck results over the randomly generated data when $N = 10$ are shown in Table 4.2. Column Obj means the average objective values solved by MIP-3 (Pearn et al., 1987), MIP-2 (Longo et al., 2006), Tabu Search (TS) (Luo et al., 2021) and our ALNS. The optimal solution can be obtained by MIP-3 and MIP-2 solved via Gurobi when the number of required edges $|\mathcal{R}|$ is 5 or 7. When $|\mathcal{R}|$ is 10, the instances become very large and cannot be solved to optimality in the limited run time of 3600s. As the objective values of MIP-3 and MIP2 being benchmark, TS and ALNS can not perform better with the average gap of 3.81% and 1.73% when $|\mathcal{R}| = 5$; the gaps of 4.65% and 2.07% when $|\mathcal{R}| = 7$; and the gaps of 4.87% and 2.62% when $|\mathcal{R}| = 10$. ALNS outperforms TS in the objective values because of the smaller gap to the optimal solutions. The computational times reveal that MIP-2 runs the fastest when the instances are small. When $|\mathcal{R}| = 10$, the average run time of TS (247.02s) and ALNS (202.99s) are much less than those of MIP. The advantage of the metaheuristic in run time becomes more obvious. Meanwhile, ALNS also outperforms TS in the run time.

One-Drone-One-Truck results over the randomly generated data when $N = 15$ are shown in Table 4.3. MIP-3 and MIP-2 solve the small instances well. However, when the number

Table 4.2: One-Drone-One-Truck Results on Randomly Generated Data $N = 10$

Instance				Obj						CPU (seconds)			
	v_t	v_d	β	MIP-3[b]	MIP-2[c]	TS	Gap%	ALNS	Gap%	MIP-3[b]	MIP-2[c]	TS	ALNS
N10E20R5	1	1	1	359.30	359.30	367.23	2.21	363.23	1.09	4.85	2.96	11.24	3.64
			2	289.67	289.67	299.32	3.33	290.62	0.33	8.25	4.90	10.63	5.72
			3	252.07	252.07	258.62	2.60	254.34	0.90	4.79	1.28	9.42	6.32
	1	2	1	358.22	358.22	371.99	3.84	365.32	1.98	4.58	0.96	4.63	11.23
			2	256.43	256.43	268.32	4.64	263.23	2.65	8.59	2.73	10.42	7.32
			3	196.29	196.29	204.23	4.05	200.42	2.11	9.12	4.36	9.98	5.43
	2	1	1	184.26	184.26	191.32	3.83	188.42	2.26	2.81	1.59	12.42	7.43
			2	165.14	165.14	173.32	4.96	168.23	1.87	2.22	2.17	9.43	5.73
			3	158.54	158.54	166.21	4.84	162.34	2.40	3.23	1.86	8.99	6.43
Ave				246.66	246.66	255.62	3.81	250.68	**1.73**	5.38	**2.53**	10.42	5.85
N10E20R7	1	1	1	462.89	462.89	483.53	4.46	469.32	1.39	37.42	7.92	5.35	16.32
			2	353.17	353.17	372.73	5.54	363.24	2.85	230.80	75.74	33.42	17.42
			3	314.13	314.13	326.42	3.91	319.40	1.68	133.66	42.36	79.54	20.32
	1	2	1	461.53	461.53	477.32	3.42	470.34	1.91	59.58	12.92	43.43	16.23
			2	319.89	319.89	335.43	4.86	326.43	2.04	497.73	126.24	63.43	34.23
			3	237.08	237.08	250.32	5.58	242.53	2.30	317.73	180.17	74.52	43.42
	2	1	1	235.27	235.27	245.32	4.27	239.43	1.77	13.81	5.16	12.43	10.32
			2	200.25	200.25	210.45	5.09	204.32	2.03	15.70	5.20	10.53	14.23
			3	193.24	193.24	202.34	4.71	198.43	2.69	18.65	4.39	7.34	15.32
Ave				308.61	308.61	322.65	4.65	314.83	**2.07**	147.23	51.12	40.00	**20.87**
N10E20R10	1	1	1	594.61[a]	594.61[a]	607.34	2.14	603.24	1.45	1443.86	1066.15	295.34	125.34
			2	439.66[a]	434.67[a]	466.34	7.29	452.32	4.06	3593.46	3144.67	222.75	220.32
			3	393.13[a]	391.29[a]	436.23	11.48	412.32	5.37	2954.81	2049.52	252.39	198.52
	1	2	1	591.06[a]	591.06[a]	603.30	2.07	599.42	1.41	1211.51	938.79	215.34	284.55
			2	426.10[a]	394.15[a]	405.63	2.91	400.23	1.54	3600.00	3600.00	199.23	204.32
			3	296.07[a]	284.21[a]	295.32	3.91	289.53	1.87	3504.08	2798.71	189.43	221.23
	2	1	1	302.16[a]	301.05[a]	308.27	2.40	304.23	1.06	1546.43	918.86	285.35	198.43
			2	258.35[a]	256.34[a]	270.43	5.50	263.23	2.69	1795.22	1008.30	263.23	174.23
			3	244.98[a]	244.32[a]	259.32	6.14	254.34	4.10	1424.13	910.31	300.11	199.99
Ave				394.01	387.97	405.80	4.87	397.65	**2.62**	2343.81	1827.58	247.02	**202.99**

[a] Not all 25 instances can be solved to optimality within the limited computational time of 3600s and the objective values are not optimal.
[b] MIP formulation are based on the VRP transformed from Pearn et al. (1987)
[c] MIP formulation are based on the VRP transformed from Longo et al. (2006)

of required edges increases, ALNS and TS perform better both in the solution quality and computational times. The objective value gaps of ALNS 1.43%, 2.39% and 2.74% show that ALNS can get an acceptable solution within up to 10.34s, 55.47s and 300.23s for $|\mathcal{R}|$ =5, 7, 10, respectively. Between these two metaheuristics, ALNS also runs faster and gets greater solutions than TS.

Table 4.3: One-Drone-One-Truck Results on Randomly Generated Data $N = 15$

Instance				Obj						CPU (seconds)			
	v_t	v_d	β	MIP-3[b]	MIP-2[c]	TS	Gap%	ALNS	Gap%	MIP-3[b]	MIP-2[c]	TS	ALNS
N15E30R5	1	1	1	446.71	446.71	454.23	1.68	450.43	0.83	3.44	1.50	12.43	6.47
			2	362.33	362.33	371.01	2.40	367.63	1.46	11.74	7.22	10.42	10.34
			3	303.35	303.35	314.42	3.65	309.34	1.98	5.32	4.00	9.45	5.63
	1	2	1	446.42	446.42	457.52	2.49	452.32	1.32	3.50	1.89	7.64	4.63
			2	340.98	340.98	348.32	2.15	344.52	1.04	16.63	10.29	8.63	7.83
			3	253.60	253.60	265.32	4.62	259.32	2.25	12.51	7.51	11.42	6.58
	2	1	1	225.79	225.79	233.43	3.39	228.32	1.12	2.77	1.47	10.24	3.21
			2	205.07	205.07	211.89	3.32	207.53	1.20	4.28	3.19	14.25	7.42
			3	190.00	190.00	200.21	5.37	193.24	1.70	2.91	2.22	11.42	5.43
Ave				308.25	308.25	317.37	3.23	312.52	**1.43**	7.01	**4.37**	10.66	6.39
N15E30R7	1	1	1	537.89	537.89	553.42	2.89	549.75	2.21	45.05	9.11	45.33	18.32
			2	424.33[a]	424.33	446.34	5.19	430.23	1.39	637.16	101.28	55.11	22.62
			3	352.69	352.69	375.43	6.45	362.34	2.74	126.60	24.21	34.63	35.23
	1	2	1	533.58	533.58	563.42	5.59	547.63	2.63	53.13	6.95	45.34	19.43
			2	399.35[a]	399.35	420.53	5.30	410.23	2.72	963.13	221.13	39.64	35.64
			3	282.83	282.83	297.34	5.13	289.53	2.37	741.21	125.79	26.71	55.47
	2	1	1	271.76	271.76	290.34	6.84	278.43	2.45	22.17	4.20	36.23	16.43
			2	239.13	239.13	250.43	4.73	244.42	2.21	72.57	16.91	53.53	20.43
			3	225.23	225.23	235.43	4.53	231.52	2.79	57.28	13.49	49.35	19.64
Ave				362.98	362.98	381.41	5.18	371.56	**2.39**	302.03	58.12	42.87	**27.02**
N15E30R10	1	1	1	665.25[a]	665.25[a]	683.24	2.70	680.34	2.27	1454.16	1203.22	320.43	290.31
			2	502.19[a]	498.74[a]	520.34	4.33	511.42	2.54	3600.00	3371.43	295.35	300.23
			3	435.41[a]	429.91[a]	450.32	4.75	440.23	2.40	2685.58	2422.05	340.52	243.52
	1	2	1	660.18[a]	660.18[a]	678.42	2.76	675.73	2.35	1305.83	1300.98	299.43	210.24
			2	481.46[a]	481.08[a]	515.62	5.00	510.23	3.90	3555.47	3600.00	287.77	199.53
			3	341.40[a]	337.74[a]	370.53	5.34	359.34	2.16	3402.47	3412.68	310.45	178.46
	2	1	1	335.85[a]	332.41[a]	359.64	6.91	348.53	3.60	1514.19	1341.94	296.34	176.34
			2	278.89[a]	278.72[a]	286.43	2.77	285.47	2.42	1454.84	1520.01	301.53	193.32
			3	262.62[a]	260.58[a]	270.31	3.73	268.34	2.98	1228.25	1015.98	296.43	200.52
Ave				440.36	441.40	459.43	4.25	453.29	**2.74**	2263.24	2132.33	305.36	**221.39**

[a] Not all 25 instances can be solved to optimality within the limited computational time of 3600s and the objective values are not optimal.
[b] MIP formulation are based on the VRP transformed from Pearn et al. (1987)
[c] MIP formulation are based on the VRP transformed from Longo et al. (2006)

4.5.1.2 Analysis on the Large-Size Instances

As shown in Table 4.2 and 4.3, Gurobi cannot solve MIP-3 and MIP-2 to optimality in 3600s and formulation can not get a feasible solution when the network is large. The two metaheuristic methods, TS and ALNS, are tested on a set of undirected rural postman problem instances (Corberán et al., 2021). The characteristic of the instances UR500 is

shown in Table 4.4. The results of four URPP500 instances are summarized in Table 4.5. The run time is limited to 1200 seconds for both TS and ALNS. ALNS is able to get better solutions for all instances with the average gap of 1.99%, 4.16%, 1.36% and 3.54%.

Table 4.4: Characteristic of Undirected Rural Postman Problem UR500

	Ave	Min	Max
Nodes	446.0	298	499
Edges	1128.9	597	1526
Req-Edges	35.3	1	99

Table 4.5: One-Drone-One-Truck Results on Large-Size Instances

v_t	v_d	β	TS	ALNS	Gap%	TS	ALNS	Gap%
				UR532			UR535	
1	1	1	10342	10034	3.07	12042	11592	3.88
		2	10225	9987	2.38	11942	11561	3.30
		3	9998	9698	3.09	10093	9899	1.96
1	2	1	8843	8733	1.26	8234	7953	3.53
		2	8632	8529	1.21	8102	7801	3.86
		3	8452	8321	1.57	7842	7504	4.50
2	1	1	7201	7033	2.40	7293	6903	5.65
		2	6992	6843	2.18	7102	6723	5.64
		3	6703	6653	0.75	6983	6643	5.12
	Ave		8599	8426	1.99	8848	8301	4.16
				UR537			UR542	
1	1	1	11023	10932	0.83	11423	11242	1.61
		2	10294	10200	0.92	11232	10923	2.83
		3	10125	10101	0.24	11001	10842	1.47
1	2	1	10023	9994	0.29	10532	10424	1.04
		2	9923	9530	4.12	10423	10211	2.08
		3	9380	9305	0.81	10232	10112	1.19
2	1	1	8990	8942	0.54	10032	9123	9.96
		2	9123	8816	3.48	9834	8942	9.98
		3	8824	8736	1.01	8988	8834	1.74
	Ave		9745	9617	1.36	10411	10073	3.54

4.5.2 One Truck and Multiple Drones

4.5.2.1 Analysis on the Small-Size Instances

Table 4.6: Two-Drones-One-Truck Results on Randomly Generated Data with $N = 10$

Instance				Obj			CPU (seconds)		
	v_t	v_d	β	TS	ALNS	Gap%	TS	ALNS	Gap%
N10E20R5	1	1	1	271.26	263.23	3.05	9.87	3.55	178.03
			2	222.32	216.62	2.63	12.01	6.07	97.86
			3	198.62	192.34	3.27	7.4	4.65	59.14
	1	2	1	280.5	275.32	1.88	13.1	4.47	193.06
			2	201.62	195.23	3.27	11.55	6.11	89.03
			3	142.67	138.42	3.07	10.83	5.07	113.61
	2	1	1	140.42	136.42	2.93	12.21	9.36	30.45
			2	114.78	110.54	3.84	8.37	5.12	63.48
			3	105.21	101.32	3.84	9.44	5.11	84.74
Ave				186.38	181.05	3.09	10.53	5.50	101.04
N10E20R7	1	1	1	418.34	409.56	2.14	33.38	17.69	88.69
			2	311.91	308.22	1.20	37.28	14.08	164.77
			3	234.7	229.68	2.19	80.68	15.62	416.52
	1	2	1	381.38	373.75	2.04	45.02	22.11	103.62
			2	259.99	253.81	2.43	64.27	39.26	63.70
			3	170.71	163.95	4.12	75.80	48.2	57.26
	2	1	1	152.8	148.61	2.82	36.51	13.5	170.44
			2	162.5	154.3	5.31	48.65	13.99	247.75
			3	126.74	121.64	4.19	28.56	20.5	39.32
Ave				246.56	240.39	2.94	50.02	22.77	150.23
N10E20R10	1	1	1	517.34	508.74	1.69	311.24	109.74	183.62
			2	488.42	471.12	3.67	235.85	222.72	5.90
			3	436.23	422.74	3.19	237.79	187.22	27.01
	1	2	1	545.42	536.02	1.75	230.34	207.35	11.09
			2	414.23	404.23	2.47	281.53	186.42	51.02
			3	335.32	324.34	3.39	286.63	234.13	22.42
	2	1	1	214.23	205.03	4.49	265.55	183.73	44.53
			2	194.22	182.13	6.64	253.03	178.93	41.41
			3	188.43	178.94	5.30	295.01	196.39	50.22
Ave				370.43	359.25	3.62	266.33	189.63	48.58

Table 4.7: Two-Drones-One-Truck Results on Randomly Generated Data with $N = 15$

Instance				Obj			CPU (seconds)		
	v_t	v_d	β	TS	ALNS	Gap%	TS	ALNS	Gap%
N15E30R5	1	1	1	327.83	312.33	4.96	14.93	7.97	87.33
			2	301.23	299.93	0.43	18.12	8.74	107.32
			3	284.32	269.74	5.41	11.85	8.53	38.92
	1	2	1	379.82	351.82	7.96	7.94	4.23	87.71
			2	309.72	299.22	3.51	11.63	6.73	72.81
			3	258.42	254.23	1.65	14.32	6.28	128.03
	2	1	1	248.83	239.32	3.97	12.24	4.61	165.51
			2	201.49	195.93	2.84	15.25	7.82	95.01
			3	189.81	185.14	2.52	11.32	4.33	161.43
Ave				277.94	267.52	3.69	13.07	6.58	104.90
N15E30R7	1	1	1	418.42	402.65	3.92	79.43	16.72	375.06
			2	392.42	385.53	1.79	46.51	24.82	87.39
			3	379.34	375.23	1.10	66.23	34.73	90.70
	1	2	1	413.92	396.23	4.46	55.24	18.93	191.81
			2	394.23	374.11	5.38	50.64	32.94	53.73
			3	372.32	345.34	7.81	65.31	53.67	21.69
	2	1	1	179.24	163.23	9.81	27.23	17.33	57.13
			2	139.53	130.92	6.58	87.63	23.33	275.61
			3	130.23	126.72	2.77	25.75	19.14	34.54
Ave				313.29	299.99	4.85	56.00	26.85	131.96
N15E30R10	1	1	1	524.64	515.44	1.78	335.53	308.53	8.75
			2	396.64	381.52	3.96	314.25	224.12	40.22
			3	343.23	333.13	3.03	323.62	226.12	43.12
	1	2	1	583.62	574.23	1.64	289.13	203.80	41.87
			2	395.23	383.43	3.08	296.37	170.21	74.12
			3	240.32	227.54	5.62	337.65	178.37	89.29
	2	1	1	218.34	213.33	2.35	315.64	225.35	40.07
			2	193.23	182.77	5.72	316.33	236.30	33.87
			3	159.32	145.34	9.62	310.43	222.19	39.71
Ave				339.40	328.53	4.09	315.44	221.66	45.67

Both ALNS and TS can solve the case when one truck and multiple drones traverse all required edges jointly. The tests are done on the randomly generated data with $|N| = 10$ and $|N| = 15$. The results are summarized in Table 4.6 and 4.7. ALNS gets better objective values with the average gaps of 3.09%, 2.94%, 3.62% and with the average run time gaps of 101.04%, 150.23% and 48.58% for $|\mathcal{N}|$ =10; with the average obj gaps of 3.69%, 4.85%, 4.09% and with the average run time gaps of 104.90%, 131.96%, 45.67% for $|\mathcal{N}|$=15. Since both ALNS and TS are limited to run in 1200s, the gaps between them show that ALNS performs better than TS for all randomly generated instances.

4.5.2.2 Analysis on the Large-Size Instances

Table 4.8: Two-Drones-One-Truck Results on Large-Size Instances

v_t	v_d	β	TS	ALNS	Gap%	TS	ALNS	Gap%
			UR532			UR535		
1	1	1	6894.67	6689.33	3.07	7528.00	7461.33	0.89
		2	6816.67	6658.00	2.38	7961.33	7728.37	3.01
		3	6665.33	6465.33	3.09	6728.67	6599.33	1.96
1	2	1	5171.33	5017.20	3.07	4760.03	4746.15	0.29
		2	5006.67	4889.00	2.41	4471.13	4355.67	2.65
		3	5034.67	4854.33	3.71	4567.00	4500.67	1.47
2	1	1	5712.67	5507.33	3.73	5793.53	5478.19	5.76
		2	5476.33	5265.00	4.01	5384.67	5092.77	5.73
		3	5059.67	4945.47	2.31	5561.65	5365.67	3.65
Ave			5759.78	5587.89	3.09	5861.78	5703.13	2.83
			UR537			UR542		
1	1	1	7348.67	7188.00	2.24	7615.33	7494.67	1.61
		2	6862.67	6789.32	1.08	7488.23	7282.76	2.82
		3	6850.00	6701.22	2.22	7334.16	7228.12	1.47
1	2	1	6315.00	6009.67	5.08	6294.33	6001.11	4.89
		2	6012.15	5836.63	3.01	6298.67	5938.76	6.06
		3	5601.37	5545.32	1.01	6088.33	5757.14	5.75
2	1	1	6725.98	6676.65	0.74	7517.00	6866.34	9.48
		2	6700.01	6668.34	0.47	7138.00	6728.34	6.09
		3	6854.67	6347.10	8.00	6912.65	6697.33	3.21
Ave			6585.61	6418.03	2.65	6965.19	6666.06	4.60

TS and ALNS methods also solve four large-size URPP500 instances when there are two drones and one truck. Both methods use the stop criterion as the maximum iteration number of 5000 and nonimproving iteration number of 1000. The objective values are shown in Table 4.8. Average gaps are 3.09%, 2.83%, 2.65% and 4.60% for instances UR532, UR535, UR537 and UR542, respectively. ALNS enables to get better solutions than TS in the situation of the multiple drones and one truck.

4.5.3 Analysis on Speed and Drone Range

The number of instances solved to optimality are given in Table 4.9. There are 25 instances of each type. All instances are solved within the limited computational time of 3600 seconds. When $|\mathcal{R}| = 5$, all 25 instances of each type can be solved optimally. When $|\mathcal{R}| = 7$, MIP-3 and MIP-2 can not obtain the optimal solutions in some cases when the truck speed $v_t = 1$, the drone speed $v_d = 2$, range $\beta = 2$ or 3. As the number of required edges $|\mathcal{R}|$ increases to 10, the property of the problem becomes obvious. The problem is the simplest to solve when the truck is faster than the drone ($v_t = 2, v_d = 1$), because the truck tends to service the most edges and the drone gets onboard the truck in the most time. Then, the problem is harder to solve when the speeds are the same ($v_t = v_d = 1$), because it leads that the problem is equivalent to 2-truck ARP with one truck having length constraint. It is the hardest to solve when the drone is faster than the truck ($v_t = 1, v_d = 2$). Because the faster drone is able to traverse more required edges and benefits in reducing the completion time.

The different maximum drone ranges also affect the complexity of the problem. For different values of $\beta = 1, 2, 3$, the average percentages of instances solved to optimality are 72.67%, 25.33%, 37.33% by MIP-3 and 82.00%, 36.67% and 56.00% by MIP-2, respectively. When the maximum flight range is short $\beta = 1$, one flight trip can not cover some edges. As the maximum range increases to $\beta = 2$, there are more feasible solutions where the drone can service some edges. If the maximum range becomes very large, the problem is equivalent to 2-truck ARP with the trucks having different speeds. That becomes easier to solve.

Table 4.9: Number of Randomly Generated Instances Solved to Optimality

			# opt			# opt	
v_t	v_d	β	Instance	MIP-3	MIP-2	Instance	MIP-3	MIP-2
1	1	1	N10E20R5	25/25	25/25	N15E30R5	25/25	25/25
		2		25/25	25/25		25/25	25/25
		3		25/25	25/25		25/25	25/25
1	2	1		25/25	25/25		25/25	25/25
		2		25/25	25/25		25/25	25/25
		3		25/25	25/25		25/25	25/25
2	1	1		25/25	25/25		25/25	25/25
		2		25/25	25/25		25/25	25/25
		3		25/25	25/25		25/25	25/25
1	1	1	N10E20R7	25/25	25/25	N15E30R7	25/25	25/25
		2		25/25	25/25		24/25	25/25
		3		25/25	25/25		25/25	25/25
1	2	1		25/25	25/25		25/25	25/25
		2		24/25	25/25		22/25	25/25
		3		24/25	24/25		25/25	25/25
2	1	1		25/25	25/25		25/25	25/25
		2		25/25	25/25		25/25	25/25
		3		25/25	25/25		25/25	25/25
1	1	1	N10E20R10	18/25	20/25	N15E30R10	18/25	19/25
		2		1/25	6/25		3/25	8/25
		3		6/25	15/25		10/25	14/25
1	2	1		19/25	21/25		20/25	20/25
		2		0/25	0/25		1/25	1/25
		3		2/25	9/25		2/25	3/25
2	1	1		17/25	22/25		17/25	21/25
		2		16/25	20/25		17/25	20/25
		3		17/25	21/25		19/25	22/25

The percentages of optimal solutions over the randomly generated instances with 10 required edges are drawn in Figure 4.4. The darker the color is, the harder the problem is solved. Thus, the worst case happens when $v_t/v_d = 0.5$ and the range parameter $\beta = 2$. Because the minimum makespan occurs in the situation where there is no or less waiting time at the drone's landing combined node.

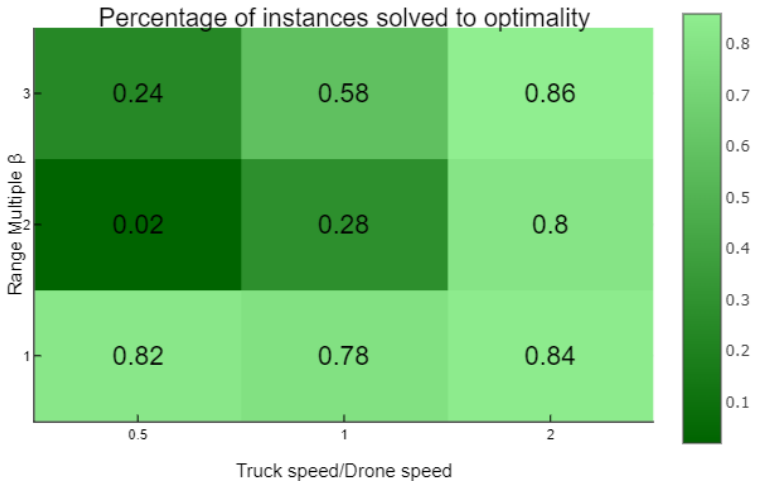


Figure 4.4: Percentage of Solutions Solved to Optimality over Randomly Generated Instances with $|\mathcal{R}| = 10$

4.5.4 Analysis on Robustness of Adaptive Large Neighborhood Search versus Tabu Search

The robustness of a metaheuristic means how much the solutions vary if being repeated several times on the same instance. The robustness of a method is expressed as the standard deviation. The randomly generated instances N15E30-R5, R7 and R10 are used to evaluate the robustness. Each instance is repeated 10 times and the standard deviation is calculated from the 10 repeated solutions. The standard deviations of ALNS and TS are illustrated in Figure 4.5. For all instances, ALNS has less Std than TS. With the increasing number of required edges, the values of Std increase. The less standard deviation is, the more stable the method is. So, ALNS has better robustness than TS.

4.6 Concluding Remarks

This chapter considers Drone-Truck Arc Routing problem. The drone and the truck cooperatively service all required edges at least once. Since the drone can fly off the road network, the DT-ARP extends the traditional ARP. With a limited battery capacity, the drone needs to fly from and to vehicles for a replacement of the battery. The key challenge is how to determine the truck and drone route to minimize the completion time. A

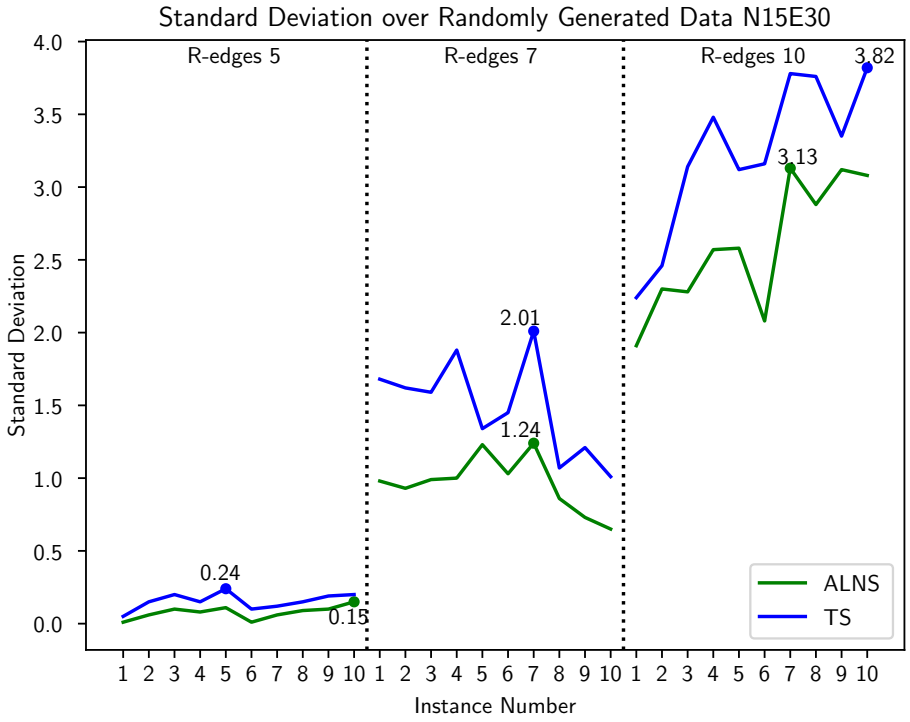


Figure 4.5: Standard Deviation of Objective Values by ALNS and TS over Randomly Generated Instances

metaheuristic method based on Adaptive Large Neighborhood Search (ALNS) is proposed to solve the Drone-Truck Arc Routing Problem (DT-ARP). The performance of ALNS is evaluated using small-size randomly generated ARP instances and large-size undirected rural postman problem instances. In order to get the optimal solution, we transform ARP into VRP with two kinds of rules and formulate a mixed-integer programming. The experiments reveal that MIP formulation can solve the problem well for the small-size network. However, for a large-size network (the number of required edges is larger than 10), an efficient and effective metaheuristic is necessary. We found that ALNS outperforms TS both in the solution quality and the computational time. The further analysis on the truck/drone speed and maximum drone flight range shows that the problem is hard to solve when the maximum flight range is double average edges' distances and the done is twice as fast as the truck. The

robustness of ALNS is also better than Tabu Search by comparing the standard deviation from the repeated solved solutions.

As for directions of future research, the metaheuristic method may be improved, such as by using strong initialization or by some other destroy and repair methods. The future work could extend to the DT-ARP with multiple trucks and multiple drones onboard per truck.

Chapter 5: Conclusion and Future Work

In this dissertation, we use an exact method and metaheuristic algorithms to solve transportation problems.

In the first problem, a leader-follower decision problem is considered in the form of bi-level optimization. In the upper level, the leader aims to minimize the total facility construction costs and hazmat exposure risks by determining facility locations and available roads for hazmat transportation. The leader affects the followers who intend to minimize their transportation costs when designing the road network. We apply a robust optimization approach to deal with the uncertainty in the exposure risk and the demand. A bi-level integer programming model is formulated where the upper level is a min-max problem and the lower level is a shortest-path problem. We devise an exact algorithm that combines a cutting plane algorithm with Benders decomposition and derive a single-level reformulation. Comparisons between two approaches are made on the Ravenna city data, in terms of objectives and the running time. The analysis on small and large size instances demonstrates that the proposed cutting plane algorithm performs much better than Gurobi as the problem size increases. The proposed cutting plane algorithm is an effective exact method for solving the robust combined facility location-network design problem.

A couple of directions for future research are suggested. First, uncertainty on origin locations can be considered. In this paper, we assume that all origin nodes are exactly known. Since the hazmat facility location problem is for long-term decision, considering new hazmat origins in the future will lead to an important problem. Second, hazmat trips to locations other than the hazmat facilities can be incorporated. Although we consider hazmat trips to hazmat facilities only in this paper, there are also hazmat trips to other destinations.

Hazmat network design policies will certainly impact not only trips to hazmat facilities, but also all other general hazmat trips. Therefore, incorporating both types of hazmat trips within a single modeling framework is a valuable research direction.

The second problem is the EV relocation and shuttle routing problem for the rebalancing operation of free-floating EV sharing systems. One of the key operational decisions for the carsharing company is how to relocate the EV fleet to meet the next day's demand with sufficient battery levels. We develop a metaheuristic ALNS algorithm for the EV relocation problem that determines where to relocate each EV and how to route the shuttles that transport the staff drivers synchronously. We apply our method to conduct numerical experiments using both randomly generated data and actual FFEVSS data in Amsterdam. We found that ALNS outperforms EBNSM both in the solution quality and the computational time. Our ALNS also produces better solutions than the RL approach but requires much longer computational time than RL. Our experiments reveal that providing the RL solution as the initial solution for ALNS is an effective and efficient solution strategy that can take advantage of both approaches, achieving the best solution quality and reducing the computational time significantly. We also demonstrate how our ALNS can be modified to solve the problem where staff drivers carry a personal mobility vehicle such as a scooter. Our further analysis provides practical recommendations on which mode of transportation will be more efficient—i.e., a small number of shuttles with large capacity or a large number of shuttles with small capacity (or even personal mobility)—in terms of total operational cost as well as wait times. Lastly, we show that our ALNS that destroys an incumbent solution partially and repairs to a new solution in each iteration is quite flexible to be applied to a dynamic environment. Specifically, our numerical results highlight the usefulness of our flexible ALNS method for an environment where some EV demands are removed or added in the course of EV relocation operations.

As directions of future research, this model can be extended for day-time static relocation. Extending this model to the 24-hour period will ordinarily require redeployment of the model

at constant, and relatively small, time intervals and also the assumption of zero new arriving demand. In that case, unlike our numerical experiments conducted with constant travel speed for the city of Amsterdam, a more robust analysis with different shuttle travel speeds can be considered to account for various traffic conditions at different times and across different locations. An important factor in the successful implementation of static repositioning is the accuracy of the demand forecast. The demand faced by a car-sharing system is highly sensitive to a variety of external factors. In this study, we base our demand forecast on past demand data on similar days and focus on synchronous modeling of relocation and routing operations. However, more sophisticated data mining models and demand prediction models can be devised.

In the final problem, Drone-Truck Arc Routing problem is studied where the drone and the truck cooperatively service all required edges at least once. Since the drone can fly off the road network, the DT-ARP extends the traditional ARP. With a limited battery capacity, the drone needs to fly from and to vehicles for a replacement of the battery. The key challenge is how to determine the truck and drone routes to minimize the completion time. A metaheuristic method based on Adaptive Large Neighborhood Search (ALNS) is proposed to solve the Drone-Truck Arc Routing Problem (DT-ARP). The performance of ALNS is evaluated using small-size randomly generated ARP instances and large-size undirected rural postman problem instances. In order to get the optimal solution, we transform the ARP into VRP with two kinds of rules and formulate mixed-integer programming. The experiments reveal that MIP formulation can solve the problem well for the small-size network. However, for a large-size network (the number of required edges is larger than 10), an efficient and effective metaheuristic is necessary. We found that ALNS outperforms Tabu Search both in the solution quality and the computational times. The further analysis on the truck/drone speed and maximum drone flight range shows that the problem is hard to solve when the maximum flight range is two times average distances of all edges and the done is twice as

fast as the truck. The robustness of ALNS is also better than Tabu Search by comparing the standard deviation from the repeated solved solutions.

As for directions of future research, the metaheuristic method may be improved, such as by using strong initialization or by some other destroy and repair methods. The future work could extend to the DT-ARP with multiple trucks and multiple drones onboard per truck.

References

Agatz, N., P. Bouman, M. Schmidt. 2018. Optimization approaches for the traveling salesman problem with drone. *Transportation Science* 52(4) 965–981.

Amaldi, E., M. Bruglieri, B. Fortz. 2011. On the hazmat transport network design problem. *Network Optimization*. Springer, 327–338.

Ardjmand, E., G. Weckman, N. Park, P. Taherkhani, M. Singh. 2015. Applying genetic algorithm to a new location and routing model of hazardous materials. *International Journal of Production Research* 53(3) 916–928.

Arslan, O., O. Jabali, G. Laporte. 2018. Exact solution of the evasive flow capturing problem. *Operations Research* 66(6) 1625–1640.

Becker, H., F. Ciari, K. W. Axhausen. 2018. Measuring the car ownership impact of free-floating car-sharing–a case study in Basel, Switzerland. *Transportation Research Part D: Transport and Environment* 65 51–62.

Benavent, E., Á. Corberán, G. Desaulniers, F. Lessard, I. Plana, J. M. Sanchis. 2014. A branch-price-and-cut algorithm for the min-max k-vehicle windy rural postman problem. *Networks* 63(1) 34–45.

Benders, J. F. 1962. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik* 4(1) 238–252.

Berglund, P. G., C. Kwon. 2014. Robust facility location problem for hazardous waste transportation. *Networks and Spatial Economics* 14(1) 91–116.

Bertsimas, D., M. Sim. 2003. Robust discrete optimization and network flows. *Mathematical Programming* 98(1-3) 49–71.

Bezanson, J., S. Karpinski, V. B. Shah, A. Edelman. 2012. Julia: A fast dynamic language for technical computing. *arXiv preprint arXiv:1209.5145* .

Bianco, L., M. Caramia, S. Giordani. 2009. A bilevel flow model for hazmat transportation network design. *Transportation Research Part C: Emerging Technologies* 17(2) 175–196.

Bianco, L., M. Caramia, S. Giordani, V. Piccialli. 2015. A game-theoretic approach for regulating hazmat transportation. *Transportation Science* 50(2) 424–438.

Bogyrbayeva, A., S. Jang, A. Shah, Y. J. Jang, C. Kwon. 2021. A reinforcement learning approach for rebalancing electric vehicle sharing systems. *IEEE Transactions on Intelligent Transportation Systems* 1–11.

Boyacı, B., K. G. Zografos, N. Geroliminis. 2015. An optimization framework for the development of efficient one-way car-sharing systems. *European Journal of Operational Research* 240(3) 718–733.

Boyacı, B., K. G. Zografos, N. Geroliminis. 2017. An integrated optimization-simulation framework for vehicle and personnel relocations of electric carsharing systems with reservations. *Transportation Research Part B: Methodological* 95 214–237.

Boysen, N., D. Briskorn, S. Fedtke, S. Schwerdfeger. 2018. Drone delivery from trucks: Drone scheduling for given truck routes. *Networks* 72(4) 506–527.

Bruglieri, M., F. Pezzella, O. Pisacane. 2017. Heuristic algorithms for the operator-based relocation problem in one-way electric carsharing systems. *Discrete Optimization* 23 56–80.

Bruglieri, M., F. Pezzella, O. Pisacane. 2019. An adaptive large neighborhood search for relocating vehicles in electric carsharing services. *Discrete Applied Mathematics* 253 185–200.

Calogiuri, T., G. Ghiani, E. Guerriero, R. Mansini. 2019. A branch-and-bound algorithm for the time-dependent rural postman problem. *Computers & Operations Research* 102 150–157.

Campbell, J. F., Á. Corberán, I. Plana, J. M. Sanchis, P. Segura. 2021. Solving the length constrained k-drones rural postman problem. *European Journal of Operational Research* 292(1) 60–72.

Campbell, J. F., Á. Corberán, I. Plana, J. M. Sanchis. 2018. Drone arc routing problems. *Networks* 72(4) 543–559.

Carotenuto, P., S. Giordani, S. Ricciardelli. 2007. Finding minimum and equitable risk routes for hazmat shipments. *Computers & Operations Research* 34(5) 1304–1327.

Cepolina, E. M., A. Farina. 2012. A new shared vehicle system for urban areas. *Transportation Research Part C: Emerging Technologies* 21(1) 230–243.

Chow, J. Y. 2016. Dynamic uav-based traffic monitoring under uncertainty as a stochastic arc-inventory routing policy. *International Journal of Transportation Science and Technology* 5(3) 167–185.

Chung, S. H., B. Sah, J. Lee. 2020. Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. *Computers & Operations Research* 123 105004.

Contreras, I., E. Fernández, G. Reinelt. 2012. Minimizing the maximum travel time in a combined model of facility location and network design. *Omega* 40(6) 847–860.

Corberán, Á., I. Plana, M. Reula, J. M. Sanchis. 2021. Arc routing problems data instances. https://www.uv.es/corberan/instancias.htm. Last updated: May 2021.

Di Febbraro, A., N. Sacco, M. Saeednia. 2012. One-way carsharing: Solving the relocation problem. *Transportation Research Record* 2319(1) 113–120.

Dille, M., S. Singh. 2013. Efficient aerial coverage search in road networks. *AIAA Guidance, Navigation, and Control (GNC) Conference*. 5094.

Dolan, E. D., J. J. Moré. 2002. Benchmarking optimization software with performance profiles. *Mathematical Programming* 91(2) 201–213.

Dunning, I., J. Huchette, M. Lubin. 2017. Jump: A modeling language for mathematical optimization. *SIAM Review* 59(2) 295–320.

Engberts, B., E. Gillissen. 2016. Policing from above: Drone use by the police. *The Future of Drone Use*. Springer, 93–113.

Erkut, E., O. Alp. 2007. Designing a road network for dangerous goods shipments. *Computers & Operations Research* 34(5) 1389–1405.

Erkut, E., S. A. Tjandra, V. Verter. 2007. Chapter 9 Hazardous Materials Transportation. *Transportation*, *Handbooks in Operations Research and Management Science*, vol. 14. Elsevier, 539–621.

Esfandeh, T., R. Batta, C. Kwon. 2017. Time-dependent hazardous-materials network design problem. *Transportation Science* 52(2) 454–473.

Firnkorn, J., M. Müller. 2011. What will be the environmental effects of new free-floating car-sharing systems? the case of car2go in ulm. *Ecological Economics* 70(8) 1519–1528.

Fontaine, P., S. Minner. 2018. Benders decomposition for the hazmat transport network design problem. *European Journal of Operational Research* 267(3) 996–1002.

Garrido, R. A. 2008. Road pricing for hazardous materials transportation in urban networks. *Networks and Spatial Economics* 8(2-3) 273–285.

Gelareh, S., D. Pisinger. 2011. Fleet deployment, network design and hub location of liner shipping companies. *Transportation Research Part E: Logistics and Transportation Review* 47(6) 947–964.

Ghaderi, A. 2015. Heuristic algorithms for solving an integrated dynamic center facility location-network design model. *Networks and Spatial Economics* 15(1) 43–69.

Ghaderi, A., M. S. Jabalameli. 2013. Modeling the budget-constrained dynamic uncapacitated facility location–network design problem and solving it via two efficient heuristics: a case study of health care. *Mathematical and Computer Modelling* 57(3-4) 382–400.

Gzara, F. 2013. A cutting plane approach for bilevel hazardous material transport network design. *Operations Research Letters* 41(1) 40–46.

Haider, Z., H. Charkhgard, S. W. Kim, C. Kwon. 2019. Optimizing the relocation operations of free-floating electric vehicle sharing systems. *Available at SSRN: http://dx.doi.org/10.2139/ssrn.3480725* .

Herrmann, S., F. Schulte, S. Voß. 2014. Increasing acceptance of free-floating car sharing systems using smart relocation strategies: a survey based study of car2go hamburg. *International Conference on Computational Logistics*. Springer, 151–162.

Hertz, A., G. Laporte, M. Mittaz. 2000. A tabu search heuristic for the capacitated arc routing problem. *Operations Research* 48(1) 129–135.

Jarboui, B., H. Derbel, S. Hanafi, N. Mladenović. 2013. Variable neighborhood search for location routing. *Computers & Operations Research* 40(1) 47–57.

Jorge, D., G. H. Correia, C. Barnhart. 2014. Comparing optimal relocation operations with simulated relocation policies in one-way carsharing systems. *IEEE Transactions on Intelligent Transportation Systems* 15(4) 1667–1675.

Kara, B. Y., V. Verter. 2004. Designing a road network for hazardous materials transportation. *Transportation Science* 38(2) 188–196.

Khoufi, I., A. Laouiti, C. Adjih. 2019. A survey of recent extended variants of the traveling salesman and vehicle routing problems for unmanned aerial vehicles. *Drones* 3(3) 66.

Killmer, K. A., G. Anandalingam, S. A. Malcolm. 2001. Siting noxious facilities under uncertainty. *European Journal of Operational Research* 133(3) 596–607.

Kortum, K., R. Schönduwe, B. Stolte, B. Bock. 2016. Free-floating carsharing: City-specific growth rates and success factors. *Transportation Research Procedia* 19 328–340.

Kwon, C., T. Lee, P. Berglund. 2013. Robust shortest path problems with two uncertain multiplicative cost coefficients. *Naval Research Logistics (NRL)* 60(5) 375–394.

Kypriadis, D., G. Pantziou, C. Konstantopoulos, D. Gavalas. 2018. Minimum walking static repositioning in free-floating electric car-sharing systems. *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 1540–1545.

Kypriadis, D., G. Pantziou, C. Konstantopoulos, D. Gavalas. 2020. Optimizing relocation cost in free-floating car-sharing systems. *IEEE Transactions on Intelligent Transportation Systems* 21(9) 4017–4030.

Laporte, G., R. Musmanno, F. Vocaturo. 2010. An adaptive large neighbourhood search heuristic for the capacitated arc-routing problem with stochastic demands. *Transportation Science* 44(1) 125–135.

Le Vine, S., J. Polak. 2019. The impact of free-floating carsharing on car ownership: Early-stage findings from london. *Transport Policy* 75 119–127.

Lenstra, J. K., A. R. Kan. 1976. On general routing problems. *Networks* 6(3) 273–280.

Leon-Blanco, J. M., P. Gonzalez-R, J. L. Andrade-Pineda, D. Canca, M. Calle. 2022. A multi-agent approach to the truck multi-drone routing problem. *Expert Systems with Applications* 195 116604.

Li, M., L. Zhen, S. Wang, W. Lv, X. Qu. 2018. Unmanned aerial vehicle scheduling problem for traffic monitoring. *Computers & Industrial Engineering* 122 15–23.

Longo, H., M. P. De Aragao, E. Uchoa. 2006. Solving capacitated arc routing problems using a transformation to the cvrp. *Computers & Operations Research* 33(6) 1823–1837.

Luo, Z., M. Poon, Z. Zhang, Z. Liu, A. Lim. 2021. The multi-visit traveling salesman problem with multi-drones. *Transportation Research Part C: Emerging Technologies* 128 103172.

Macrina, G., L. D. P. Pugliese, F. Guerriero, G. Laporte. 2020. Drone-aided routing: A literature review. *Transportation Research Part C: Emerging Technologies* 120 102762.

Marcotte, P., A. Mercier, G. Savard, V. Verter. 2009. Toll policies for mitigating hazardous materials transport risk. *Transportation Science* 43(2) 228–243.

Mattia, G., R. G. Mugion, L. Principato. 2019. Shared mobility as a driver for sustainable consumptions: The intention to re-use free-floating car sharing. *Journal of Cleaner Production* 237 117404.

Melkote, S., M. S. Daskin. 2001a. Capacitated facility location/network design problems. *European Journal of Operational Research* 129(3) 481–495.

Melkote, S., M. S. Daskin. 2001b. An integrated model of facility location and transportation network design. *Transportation Research Part A: Policy and Practice* 35(6) 515–538.

Mogili, U. R., B. Deepak. 2018. Review on application of drone systems in precision agriculture. *Procedia Computer Science* 133 502–509.

Monroy-Licht, M., C. A. Amaya, A. Langevin. 2017. Adaptive large neighborhood search algorithm for the rural postman problem with time windows. *Networks* 70(1) 44–59.

Noel, L., G. Z. de Rubens, B. K. Sovacool, J. Kester. 2019. Fear and loathing of electric vehicles: the reactionary rhetoric of range anxiety. *Energy Research & Social Science* 48 96–107.

Nourinejad, M., M. J. Roorda. 2015. Carsharing operations policies: a comparison between one-way and two-way systems. *Transportation* 42(3) 497–518.

Nourinejad, M., S. Zhu, S. Bahrami, M. J. Roorda. 2015. Vehicle relocation and staff rebalancing in one-way carsharing systems. *Transportation Research Part E: Logistics and Transportation Review* 81 98–113.

Oh, H., S. Kim, A. Tsourdos, B. A. White. 2014. Coordinated road-network search route planning by a team of uavs. *International Journal of Systems Science* 45(5) 825–840.

Oh, H., H. Shin, A. Tsourdos, B. White, P. Silson. 2011. Coordinated road network search for multiple uavs using dubins path. *Advances in Aerospace Guidance, Navigation and Control*. Springer, 55–65.

Otto, A., N. Agatz, J. Campbell, B. Golden, E. Pesch. 2018. Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. *Networks* 72(4) 411–458.

Pearn, W.-L., A. Assad, B. L. Golden. 1987. Transforming arc routing into node routing problems. *Computers & Operations Research* 14(4) 285–288.

Preeti, W., S. Prasenjit. 2019. Car sharing market size by model (p2p, station-based, free-floating), by business model (round trip, one way), by application (business, private), industry analysis report, regional outlook, application potential, price trend, competitive market share & forecast, 2020 – 2026. https://www.gminsights.com/industry-analysis/carsharing-market.

Rabbani, M., R. Heidari, H. Farrokhi-Asl, N. Rahimi. 2018. Using metaheuristic algorithms to solve a multi-objective industrial hazardous waste location-routing problem considering incompatible waste types. *Journal of Cleaner Production* 170 227–241.

Rabta, B., C. Wankmüller, G. Reiner. 2018. A drone fleet model for last-mile distribution in disaster relief operations. *International Journal of Disaster Risk Reduction* 28 107–112.

Rahmaniani, R., A. Ghaderi. 2013. A combined facility location and network design problem with multi-type of capacitated links. *Applied Mathematical Modelling* 37(9) 6400–6414.

Rakha, T., A. Gorodetsky. 2018. Review of unmanned aerial system (uas) applications in the built environment: Towards automated building inspection procedures using drones. *Automation in Construction* 93 252–264.

Ravi, R., A. Sinha. 2006. Approximation algorithms for problems combining facility location and network design. *Operations Research* 54(1) 73–81.

Roberti, R., M. Ruthmair. 2021. Exact methods for the traveling salesman problem with drone. *Transportation Science* 55(2) 315–335.

Romero, N., L. K. Nozick, N. Xu. 2016. Hazmat facility location and routing analysis with explicit consideration of equity using the gini coefficient. *Transportation Research Part E: Logistics and Transportation Review* 89 165–181.

Ropke, S., D. Pisinger. 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science* 40(4) 455–472.

Samanlioglu, F. 2013. A multi-objective mathematical model for the industrial hazardous waste location-routing problem. *European Journal of Operational Research* 226(2) 332–340.

Santos, G., G. Correia. 2015. A mip model to optimize real time maintenance and relocation operations in one-way carsharing systems. *Transportation Research Procedia* 10 384–392.

Shaheen, S. A., A. P. Cohen. 2007. Growth in worldwide carsharing: An international comparison. *Transportation Research Record* 1992(1) 81–89.

Sharma, S., S. V. Ukkusuri, T. V. Mathew. 2009. Pareto optimal multiobjective optimization for robust transportation network design problem. *Transportation Research Record* 2090(1) 95–104.

Shaw, P. 1998. Using constraint programming and local search methods to solve vehicle routing problems. *International Conference on Principles and Practice of Constraint Programming*. Springer, 417–431.

Sun, L., M. H. Karwan, C. Kwon. 2015. Robust hazmat network design problems considering risk uncertainty. *Transportation Science* 50(4) 1188–1203.

Sun, L., M. H. Karwan, C. Kwon. 2017. Generalized bounded rationality and robust multicommodity network design. *Operations Research* 66(1) 42–57.

Tagmouti, M., M. Gendreau, J.-Y. Potvin. 2010. A variable neighborhood descent heuristic for arc routing problems with time-dependent service costs. *Computers & Industrial Engineering* 59(4) 954–963.

Tagmouti, M., M. Gendreau, J.-Y. Potvin. 2011. A dynamic capacitated arc routing problem with time-dependent service costs. *Transportation Research Part C: Emerging Technologies* 19(1) 20–28.

Taslimi, M., R. Batta, C. Kwon. 2017. A comprehensive modeling framework for hazmat network design, hazmat response team location, and equity of risk. *Computers & Operations Research* 79 119–130.

United Nations. Committee of Experts on the Transport of Dangerous Goods. 2009. *Recommendations on the transport of dangerous goods: model regulations*, vol. 2. United Nations Publications.

U.S. Bureau of Labor Statistics. 2021. Occupational Outlook Handbook, Passenger Vehicle Drivers. https://www.bls.gov/ooh/transportation-and-material-moving/passenger-vehicle-drivers.htm.

U.S. Department of Transportation. 2015. Commodity Flow Survey: United States: 2012, Hazardous Materials. https://www.census.gov/library/publications/2015/econ/ec12tcf-us-hm.html/. Accessed February, 2015.

Verter, V., B. Y. Kara. 2008. A path-based approach for hazmat transport network design. *Management Science* 54(1) 29–40.

Vincent, F. Y., S.-W. Lin. 2015. Iterated greedy heuristic for the time-dependent prize-collecting arc routing problem. *Computers & Industrial Engineering* 90 54–66.

Wang, J., Y. Kang, C. Kwon, R. Batta. 2012. Dual toll pricing for hazardous materials transport with linear delay. *Networks and Spatial Economics* 12(1) 147–165.

Wang, Z., J.-B. Sheu. 2019. Vehicle routing problem with drones. *Transportation Research Part B: Methodological* 122 350–364.

Weikl, S., K. Bogenberger. 2013. Relocation strategies and algorithms for free-floating car sharing systems. *IEEE Intelligent Transportation Systems Magazine* 5(4) 100–111.

Weikl, S., K. Bogenberger. 2015. A practice-ready relocation model for free-floating carsharing systems with electric vehicles–mesoscopic approach and field trial results. *Transportation Research Part C: Emerging Technologies* 57 206–223.

Wielinski, G., M. Trépanier, C. Morency. 2015. What about free-floating carsharing? a look at the montreal, canada, case. *Transportation Research Record* 2563(1) 28–36.

Xie, Y., W. Lu, W. Wang, L. Quadrifoglio. 2012. A multimodal location and routing model for hazardous materials transportation. *Journal of Hazardous Materials* 227 135–141.

Xin, C., L. Qingge, J. Wang, B. Zhu. 2015. Robust optimization for the hazardous materials transportation network design problem. *Journal of Combinatorial Optimization* 30(2) 320–334.

Yu, L., E. Yang, P. Ren, C. Luo, G. Dobie, D. Gu, X. Yan. 2019. Inspection robots in oil and gas industry: a review of current solutions and future trends. *2019 25th International Conference on Automation and Computing (ICAC)*. IEEE, 1–6.

Zhao, M., X. Li, J. Yin, J. Cui, L. Yang, S. An. 2018. An integrated framework for electric vehicle rebalancing and staff relocation in one-way carsharing systems: Model formulation and lagrangian relaxation-based solution approach. *Transportation Research Part B: Methodological* 117 542–572.

Appendix A: Copyright Permissions

The permission below is for the use of material in Chapter 2.

PUBLISHING AGREEMENT

In order to ensure both the widest dissemination and protection of material published in our Journal, we require Authors to execute an author agreement in writing with Institute of Industrial and Systems Engineers (IISE) (hereinafter 'the Society') for the rights of copyright for the Articles they contribute. This enables our Publisher, on behalf of IISE, to ensure protection against infringement.

This is an agreement under which you, the author, assign copyright in your article to the Society to allow us to publish your article, including abstract, tables, figures, data, and supplemental material hosted by us, as the Version of Record (VoR) in the Journal for the full period of copyright throughout the world, in all forms and all media, subject to the Terms & Conditions below.

Article (the "Article") entitled:	Exact Robust Solutions for the Combined Facility Location and Network Design Problem in Hazardous Materials Transportation
Article DOI:	10.1080/24725854.2019.1697017
Author(s):	Xufei Liu, Changhyun Kwon
To publish in the Journal:	IISE Transactions
Journal ISSN:	2472-5862

STATEMENT OF ORIGINAL COPYRIGHT OWNERSHIP / CONDITIONS

In consideration of the publication of the Article, you hereby grant with full title guarantee all rights of copyright and related rights in the above specified Article as the Version of Scholarly Record which is intended for publication in all forms and all media (whether known at this time or developed at any time in the future) throughout the world, in all languages, for the full term of copyright, to take effect if and when the Article is accepted for publication in the Journal.

ASSIGNMENT OF PUBLISHING RIGHTS

I hereby assign the Society with full title guarantee all rights of copyright and related publishing rights in my article, in all forms and all media (whether known at this time or developed at any time in the future) throughout the world, in all languages, where our rights include but are not limited to the right to translate, create adaptations, extracts, or derivative works and to sub-license such rights, for the full term of copyright (including all renewals and extensions of that term), to take effect if and when the article is accepted for publication. If a statement of government or corporate ownership appears above, that statement modifies this assignment as described.

I confirm that I have read and accept the full Terms & Conditions below including my author warranties, and have read and agree to comply with the Journal's policies on peer review and publishing ethics.

Signed and dated: **Changhyun Kwon, 21 November 2019**

IISE, 21 November 2019

THIS FORM WILL BE RETAINED BY THE PUBLISHER.

ASSIGNMENT OF COPYRIGHT: TERMS & CONDITIONS

DEFINITION

1. Your article is defined as comprising (a) your Accepted Manuscript (AM) in its final form; (b) the final, definitive, and citable Version of Record (VoR) including the abstract, text, bibliography, and all accompanying tables, illustrations, data, and media; and (c) any supplemental material hosted by Taylor & Francis and/or the Society. This assignment and these Terms & Conditions constitute the entire agreement and the sole understanding between you and us ('agreement'); no amendment, addendum, or other communication will be taken into account when interpreting your and our rights and obligations under this agreement, unless amended by a written document signed by both of us.

TAYLOR &FRANCIS' RESPONSIBILITIES

2. If deemed acceptable by the Editors of the Journal, we shall prepare and publish your article in the Journal. We may post your accepted manuscript in advance of the formal publication of the VoR. We reserve the right to make such editorial changes as may be necessary to make the article suitable for publication, or as we reasonably consider necessary to avoid infringing third-party rights or breaching any laws; and we reserve the right not to proceed with publication for whatever reason.
3. Taylor & Francis will deposit your Accepted Manuscript (AM) to any designated institutional repository including PubMedCentral (PMC) with which Taylor & Francis has an article deposit agreement; see 4 iv (a) below.

RIGHTS RETAINED BY YOU AS AUTHOR

4. These rights are personal to you, and your co-authors, and cannot be transferred by you to anyone else. Without prejudice to your rights as author set out below, you undertake that the fully reference-linked Version of Record (VOR) will not be published elsewhere without our prior written consent. You assert and retain the following rights as author(s):
    i. The right to be identified as the author of your article, whenever and wherever the article is published, as defined in US Law 94-553 (Copyright Act) and, so far as is legally possible, any corresponding rights we may have in any territory of the world.
    ii. The right to retain patent rights, trademark rights, or rights to any process, product or procedure described in your article.
    iii. The right to post and maintain at any time the Author's Original Manuscript (AOM; your manuscript in its original and unrefereed form; a 'preprint').
    iv. The right to post at any time after publication of the VoR your AM (your manuscript in its revised after peer review and accepted for publication form; a 'postprint') as a digital file on your own personal or departmental website, provided that you do not use the VoR published by us, and that you include any amendments or deletions or warnings relating to the article issued or published by us; and with the acknowledgement: 'The Version of Record of this manuscript has been published and is available in <JOURNAL TITLE> <date of publication> http://www.tandfonline.com/<Article DOI>.'
        a. Please note that embargoes apply with respect to posting the AM to an institutional or subject repository. For further information, please see our list of journals with applicable embargo periods: PDF | Excel. For the avoidance of doubt, you are not permitted to post the final published paper, the VoR published by us, to any site, unless it has been published as Open Access on our website.
        b. If, following publication, you or your funder pay an Article Publishing Charge for retrospective Open Access publication, you may then opt for one of three licenses: CC BY, CC BY-NC, or CC BY-NC-ND; if you do not respond, we shall assign a CC BY licence. All rights in the article will revert to you as author.
    v. The right to share with colleagues copies of the article in its published form as supplied to you by Taylor & Francis as a digital eprint or printed reprint on a non-commercial basis.
    vi. The right to make printed copies of all or part of the article on a non-commercial basis for use by you for lecture or classroom purposes provided that such copies are not offered for sale or distributed in any systematic way, and provided that acknowledgement to prior publication in the Journal is given.
    vii. The right, if the article has been produced within the scope of your employment, for your employer to use all or part of the article internally within the institution or company on a non-commercial basis provided that acknowledgement to prior publication in the Journal is given.
    viii. The right to include the article in a thesis or dissertation that is not to be published commercially, provided that acknowledgement to prior publication in the Journal is given.
    ix. The right to present the article at a meeting or conference and to distribute printed copies of the article to the delegates attending the meeting provided that this is not for commercial purposes and provided that acknowledgement to prior publication in the Journal is given.
    x. The right to use the article in its published form in whole or in part without revision or modification in personal compilations, or other publications of your own work, provided that acknowledgement to prior publication in the Journal is given.
    xi. The right to expand your article into book-length form for publication provided that acknowledgement to prior publication in the Journal is made explicit (see below). Where permission is sought to re-use an article in a book chapter or edited collection on a commercial basis a fee will be due, payable by the publisher of the new work. Where you as the author of the article have had the lead role in the new work (i.e., you are the author of the new work or the editor of the edited collection), fees will be waived. Acknowledgement to prior publication in the Journal should be made explicit (see below):

        Acknowledgement: This <chapter or book> is derived in part from an article published in <JOURNAL TITLE> <date of publication> <copyright <the Society>, available online: http://www.tandfonline.com/<Article DOI>

If you wish to use your article in a way that is not permitted by this agreement, please contact permissionrequest@tandf.co.uk

WARRANTIES MADE BY YOU AS AUTHOR

5. You warrant that:
    i. All persons who have a reasonable claim to authorship are named in the article as co-authors including yourself, and you have not fabricated or misappropriated anyone's identity, including your own.

ii. You have been authorized by all such co-authors to sign this agreement as agent on their behalf, and to agree on their behalf the priority of the assertion of copyright and the order of names in the publication of the article.
iii. The article is your original work, apart from any permitted third-party copyright material you include, and does not infringe any intellectual property rights of any other person or entity and cannot be construed as plagiarizing any other published work, including your own published work.
iv. The article is not currently under submission to, nor is under consideration by, nor has been accepted by any other journal or publication, nor has been previously published by any other journal or publication, nor has been assigned or licensed by you to any third party.
v. The article contains no content that is abusive, defamatory, libelous, obscene, fraudulent, nor in any way infringes the rights of others, nor is in any other way unlawful or in violation of applicable laws.
vi. Research reported in the article has been conducted in an ethical and responsible manner, in full compliance with all relevant codes of experimentation and legislation. All articles which report in vivo experiments or clinical trials on humans or animals must include a written statement in the Methods section that such work was conducted with the formal approval of the local human subject or animal care committees, and that clinical trials have been registered as applicable legislation requires.
vii. Any patient, service user, or participant (or that person's parent or legal guardian) in any research or clinical experiment or study who is described in the article has given written consent to the inclusion of material, text or image, pertaining to themselves, and that they acknowledge that they cannot be identified via the article and that you have anonymized them and that you do not identify them in any way. Where such a person is deceased, you warrant you have obtained the written consent of the deceased person's family or estate.
viii. You have complied with all mandatory laboratory health and safety procedures in the course of conducting any experimental work reported in your article; your article contains all appropriate warnings concerning any specific and particular hazards that may be involved in carrying out experiments or procedures described in the article or involved in instructions, materials, or formulae in the article; your article includes explicitly relevant safety precautions; and cites, if an accepted Standard or Code of Practice is relevant, a reference to the relevant Standard or Code.
ix. You have acknowledged all sources of research funding, as required by your research funder, and disclosed any financial interest or benefit you have arising from the direct applications of your research.
x. You have obtained the necessary written permission to include material in your article that is owned and held in copyright by a third party, which shall include but is not limited to any proprietary text, illustration, table, or other material, including data, audio, video, film stills, screenshots, musical notation and any supplemental material.
xi. You have read and complied with our policy on publishing ethics.
xii. You have read and complied with the Journal's Instructions for Authors.
xiii. You have read and complied with our guide on peer review.
xiv. You will keep us and our affiliates indemnified in full against all loss, damages, injury, costs and expenses (including legal and other professional fees and expenses) awarded against or incurred or paid by us as a result of your breach of the warranties given in this agreement.
xv. You consent to allowing us to use your article for marketing and promotional purposes.

GOVERNING LAW

6. This agreement (and any dispute, proceeding, claim or controversy in relation to it) is subject to US copyright laws.

Appendix B: Mathematical Models of Chapter 2

B.1 Single-Level Robust Facility Location Problem

$$\underset{\mathbf{y}}{\text{minimize}} \left[w_1 \sum_{i \in \mathcal{M}} F_i y_i + w_2 \Big(\sum_{(i,j) \in \mathcal{A}} \sum_{s \in \mathcal{S}} N^s R^s_{ij} x^s_{ij} + \sum_{s \in \mathcal{S}} \rho^s + \sum_{(i,j) \in \mathcal{A}} \xi_{ij} + \Gamma_u \theta_u + \Gamma_v \theta_v \Big) \right]$$

$$\text{subject to} \quad \sum_{(i,j) \in \mathcal{A}} x^s_{ij} - \sum_{(j,i) \in \mathcal{A}} x^s_{ji} \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -y_i & \text{if } i \in \mathcal{M} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}$$

$$c_{ij} - \zeta^s_i + \zeta^s_j - \phi^s_{ij} = 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\phi^s_{ij} \leq M(1 - x^s_{ij}) \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\zeta^s_i \leq M[1 - (\sum_{(i,j) \in \mathcal{A}} x^s_{ij} - \sum_{(j,i) \in \mathcal{A}} x^s_{ji} + y_i)] \qquad \forall i \in \mathcal{M}, s \in \mathcal{S}$$

$$\rho^s - \sum_{(i,j) \in \mathcal{A}} \eta^s_{ij} + \theta_u \geq \sum_{(i,j) \in \mathcal{A}} K^s R^s_{ij} x^s_{ij} \qquad \forall s \in \mathcal{S}$$

$$\xi_{ij} - \sum_{s \in \mathcal{S}} \pi^s_{ij} + \theta_v \geq \sum_{s \in \mathcal{S}} N^s Q^s_{ij} x^s_{ij} \qquad \forall (i,j) \in \mathcal{A}$$

$$\eta^s_{ij} + \pi^s_{ij} \geq K^s Q^s_{ij} x^s_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$x^s_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$y_i \in \{0,1\} \qquad \forall i \in \mathcal{M}$$

$$\zeta^s_i \geq 0 \qquad \forall i \in \mathcal{M}, s \in \mathcal{S}$$

$$\zeta^s_i \text{ free} \qquad \forall i \notin \mathcal{M}, s \in \mathcal{S}$$

$$\phi^s_{ij}, \rho^s, \xi_{ij}, \eta^s_{ij}, \pi^s_{ij}, \theta_u, \theta_v \geq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

B.2 Single-Level Robust Network Design Problem

$$\underset{\mathbf{x},\mathbf{z}}{\text{minimize}} \left[\sum_{(i,j)\in\mathcal{A}} \sum_{s\in\mathcal{S}} N^s R^s_{ij} x^s_{ij} + \sum_{s\in S} \rho^s + \sum_{(i,j)\in\mathcal{A}} \xi_{ij} + \Gamma_u \theta_u + \Gamma_v \theta_v \right]$$

$$\text{subject to} \quad \sum_{(i,j)\in\mathcal{A}} x^s_{ij} - \sum_{(j,i)\in\mathcal{A}} x^s_{ji} \begin{cases} = 1 & \text{if } i = o(s) \\ \geq -1 & \text{if } i \in \mathcal{K} \\ = 0 & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{N}, s \in \mathcal{S}$$

$$x^s_{ij} \leq z_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$c_{ij} - \zeta^s_i + \zeta^s_j + \mu^s_{ij} - \phi^s_{ij} = 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\phi^s_{ij} \leq M(1 - x^s_{ij}) \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\zeta^s_i \leq M[1 - (\sum_{(i,j)\in\mathcal{A}} x^s_{ij} - \sum_{(j,i)\in\mathcal{A}} x^s_{ji} + 1)] \qquad \forall i \in \mathcal{K}, s \in \mathcal{S}$$

$$\mu^s_{ij} \leq M[1 - (-x^s_{ij} + z_{ij})] \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$\rho^s - \sum_{(i,j)\in\mathcal{A}} \eta^s_{ij} + \theta_u \geq \sum_{(i,j)\in\mathcal{A}} K^s R^s_{ij} x^s_{ij} \qquad \forall s \in \mathcal{S}$$

$$\xi_{ij} - \sum_{s\in\mathcal{S}} \pi^s_{ij} + \theta_v \geq \sum_{s\in\mathcal{S}} N^s Q^s_{ij} x^s_{ij} \qquad \forall (i,j) \in \mathcal{A}$$

$$\eta^s_{ij} + \pi^s_{ij} \geq K^s Q^s_{ij} x^s_{ij} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$x^s_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$

$$z_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}$$

$$\zeta^s_i \geq 0 \qquad \forall i \in \mathcal{K}, s \in \mathcal{S}$$

$$\zeta^s_i \text{ free} \qquad \forall i \notin \mathcal{K}, s \in \mathcal{S}$$

$$\mu^s_{ij}, \phi^s_{ij}, \rho^s, \xi_{ij}, \eta^s_{ij}, \pi^s_{ij}, \theta_u, \theta_v \geq 0 \qquad \forall (i,j) \in \mathcal{A}, s \in \mathcal{S}$$