Optimization of passenger and freight co-transportation in modular transit systems

Chenwei Peng

Delft University of Technology

TU Delft

Optimization of passenger and freight co-transportation in modular transit systems

by

Chenwei Peng

Instructor: Dongyang Xia, Yahan Lu, Sh. Sharif Azadeh, Yousef Maknoon
Project Duration: 2. 2025 - 9. 2025
Faculty: Civil Engineering and Geosciences, Delft

TU Delft

Summary

Urban public transportation often exhibits pronounced spatio-temporal imbalances in passenger demand, resulting in capacity shortages during peak periods and excessive idle capacity during off-peak times. Meanwhile, with the expansion of e-commerce, urban freight demand continues to grow, making the use of idle public transportation space for freight transport an ideal approach. This study proposes an optimization framework for integrated passenger-freight co-transportation using Modular Autonomous Vehicles (MAVs), formulating a Mixed-Integer Quadratically Constrained Programming (MIQCP) Path-based model based on a space-time network to address the Modular Autonomous Unit (MAU) Routing Problem. The model integrates Fixed-Route Transit (FRT) and Demand-Responsive Transit (DRT), allowing MAUs to dynamically couple/decouple across different routes to meet the spatio-temporal demands of passengers and freight, with the objective of minimizing total operational costs. To tackle the computational complexity of large-scale instances, a customized Adaptive Large Neighborhood Search (ALNS) algorithm is designed, incorporating two initial solution generation methods (GUROBI and Greedy heuristic) and iteratively optimizing solutions through destroy and repair operations. A real-world case study based on the Shanghai bus network validates the effectiveness of the proposed approach. The results demonstrate that the MAU co-transportation system can effectively utilize vehicle compartment space to simultaneously transport passengers and freight, significantly reducing empty load rates, leading to a substantial reduction in operating costs. Without using the co-transportation mode, the number of MAUs used would increase significantly, accompanied by a 3.9% cost increase. Compared to the traditional combination of public transit and delivery vans, costs are reduced by 83.4%. This co-transportation modular transit system, with its unique flexibility, can provide efficient and low-cost transportation services for both passengers and freight within cities.

Contents

1

Introduction

Current urban public transportation systems typically use buses operating along fixed routes and on fixed schedules. Meanwhile, passenger demand fluctuates in time and space, such as peak and off-peak hours, and between residential and office areas. Often during off-peak hours, passenger demand decreases significantly, and buses can often be observed to have a large amount of wasted capacity. In recent years, urban freight demand has continued to grow alongside the expansion of e-commerce. For operators, there is a growing interest in utilizing the excess capacity of buses for freight transport. This trend of passenger and freight co-transportation brings new opportunities and challenges to improving the efficiency of urban transportation systems.

There are many difficulties in building a co-transportation system. Firstly, daily fluctuations in urban passenger demand make it challenging to balance public transport operating costs and passenger waiting times (Shi and X. Li 2021). During off-peak hours when passenger demand is low, maintaining the same bus arrival frequency as during peak periods leads to low space utilization and increased operating costs. However, reducing bus frequencies would result in longer waiting times for passengers, thereby decreasing service satisfaction. At the same time, since freight vehicles are restricted by emission zones and time windows, the cost of freight receiving and delivery services in the cities is currently high. In complex urban road environments, integrating passenger and freight transport raises capacity allocation, route planning, and scheduling issues. To address these challenges, researchers are continuously exploring new technologies and flexible transport solutions.

In the early stages of the research, Machado, Pimentel, and Sousa (2023) proposed retrofitting some buses so that part of their interior space is used for passenger transport while the remaining space is allocated for freight transportation. However, this modification still struggles to serve as an optimal solution due to the fixed passenger and freight space allocation. Modular Autonomous Vehicle (MAV) can reduce capacity redundancy since the capacity of a single Modular Autonomous Unit (MAU) is smaller than that of a traditional bus. However, as a product of recent advancements in autonomous driving technology, research on MAUs is still in its early stages, with most studies focusing solely on passenger demand and operating on a single bus line. For integrated passenger-freight transport systems, current research primarily focuses on transportation along fixed lines (Lin and F. Zhang 2024) or treats MAUs as taxis capable of transporting freight (Hatzenbühler et al. 2024), starting from a depot to execute a series of pre-booked tasks before returning to the origin. These studies on passenger and freight intermodal transport of MAUs often ignore the transportation network formed by multiple routes and their dynamic coordination. This study developed a system model for passenger and freight co-transportation to address the research gap by completing the routing allocation for each MAU on the network.

NExT Company has currently designed a mature MAV system (NEXT Modular Vehicles 2024), as shown in Figure 1.1, which can achieve coupling/decoupling while in motion and can open front and rear carriage doors when coupled. This function ensures that passengers can freely move between multiple coupled MAUs, making transfers between different routes more convenient. Additionally, NExT

Company has proposed a feature that allows freight to be transferred to different MAUs through automated machinery, enabling more flexible freight pick-up and delivery across different routes by moving freight between modules when MAUs are coupled. One of the important advantage of NExT Company's MAV over traditional public transportation is that the autonomous driving feature reduces labor costs, with only energy costs during operation, and provides greater flexibility. It can precisely fulfill passenger demands when they arise, controlling passenger waiting time at stations. Meanwhile, when there is spare capacity in the carriage, it can meet freight demands within designated time windows.

Figure 1.1: NExT Company's MAV

This study proposes a MAV Routing Problem and models the problem as a Mixed-Integer Quadratically Constrained Programming (MIQCP) model based on a Space-Time Network, integrating autonomous units and paths. The Space-Time Network includes depots and stops at all discrete timestamps, connected by Travel arcs, Delivery arcs, Transfer arcs, and Waiting arcs to form paths. The MAU in this study is designed to function both as a bus, transporting passengers and freight along fixed bus lines, and as a delivery vehicle, transporting freight between pick-up and delivery stops outside the lines. Additionally, each MAU is capable of operating across multiple lines. The model aims to find the minimum cost required for MAUs to fulfill all transportation demands. The results display the transportation paths for each MAU and the allocation of each transportation demand. A customized Adaptive Large Neighborhood Search (ALNS) algorithm is employed to compute the results, with its quality compared against exact solutions obtained from the GUROBI solver in small-scale tests. Two initial solution methods are used: one selects a small number of demand-containing paths for GUROBI computation, and the other applies a Greedy heuristic method to quickly assign all demands to MAUs. The ALNS iteratively destroys and reconstructs parts of the current solution to find better solutions. The destruction and repair processes are divided into two levels: Path-level, involving large-scale deletion and reconstruction of entire paths, and Arc-level, involving deconstruction and reconstruction of selected paths. The algorithm is customized with multiple operators tailored to the MAV Routing Problem, including destroy operators (Path-level: Random Destroy, Worst Cost Destroy, Demand-based Destroy; Arc-level: Arc Removal Destroy, Arc Search Destroy) and repair operators (Path-level: Random Repair, Greedy Repair, Utility Maximization Repair; Arc-level: Arc Insertion Repair, Chain Repair, Hybrid Repair), with operator weights dynamically adjusted using Roulette Wheel Selection. The ALNS uses Simulated Annealing (SA) as the acceptance criterion, accepting not only improved solutions but also worse solutions with a certain probability to escape local optima. By dynamically adjusting the destruction rate and operator scoring mechanism, this ALNS achieves a balance between local optimization depth and global search breadth. Daily passenger demand at stations along fixed bus lines can be accurately estimated for specific time periods (Zhou et al. 2016), and since that the bus timetable is known, the vast majority of passengers are assumed to arrive at stations according to the timetable, reducing their waiting time. Additionally, pick-up and delivery demands for freight can be predicted in advance using GPS positioning technology (Holguín-Veras, Amaral, and Rivera-Gonzalez 2024). By obtaining real data from Shanghai, this study verified the model and algorithm using real cases. This enables the models and algorithms of this study to be widely used in modular transit systems for passenger and

freight co-transportation in various cities.

The structure of the remainder of this thesis is as follows. Section 2 discusses the research questions of this study and outlines the approach to addressing them. Section 3 reviews related literature and summarizes the contributions of this study. Section 4 provides the problem description. In Section 5, a Mixed-Integer Quadratically Constrained Programming (MIQCP)model for the MAV Routing Problem is established. Then, in Section 6, a customized ALNS algorithm is designed, and its operational process is detailed. Section 7 presents the computational results, comparing the accuracy and computational time of different methods, and includes a case study based on real-world bus lines and pick-up and delivery stops in Shanghai. Section 8 offers conclusions and directions for future research.

2

Research questions

To establish a passenger and freight co-transportation system using MAUs within cities, the following core questions need to be answered:

Is passenger and freight co-transportation in modular transit systems worthy of large-scale application within cities?

The main research question is answered by the following sub-questions. Table 2.1 shows these sub-questions and provides solutions:

Table 2.1: Sub-questions for research and corresponding solutions

Sub-questions	Methods
1) What aspects does the network of this study consider?	Space-Time Network
2) What kind of mathematical model is best suited to solve this MAU Routing Problem?	Path-based Mixed Integer Linear Programming
3) What methods are used to solve the optimization model?	GUROBI and Heuristic Method (Adaptive Large Neighborhood Search)
4) How do solution methods perform at different scales?	Numerical experiments based on simulated data
5) Compared with separate modular transit system and traditional transportation system, how big is the advantage of MAV?	Numerical experiments based on real data

This research examines the necessity and feasibility of establishing a passenger and freight co-transportation modular transit system within cities to improve transportation efficiency and cost. Due to the complexity of the problem, large-scale solvers often struggle to find high-quality solutions within an acceptable time frame. Heuristic algorithms offer the potential for greater computational efficiency. In general, these research questions aim to demonstrate the significant potential of this system in terms of flexibility, space utilization, and sustainability, making it a worthy candidate for development as a new type of transportation system.

3

Literature review

At present, the research on the Passenger-Freight intermodal transport problem of Modular Autonomous Vehicles (MAV) is in its infancy, and there are still many gaps to be explored. Therefore, this section mainly reviews the construction of Space-Time networks in the Passenger-Freight intermodal transport problem of other transportation modes and the current research status of MAVs.

3.1. Passenger-Freight intermodal transport based on the Space-Time network

Passenger-Freight intermodal transport has been used for decades for long-haul transport, such as air transport (Mason 1967) and rail transport (Yang, Xie, and Wang 2024). The modeling approach of constructing a Space-Time network to optimize the shared capacity of the two services can integrate different modes of transportation, support the dynamic changes in passenger and cargo demand, and flexibly adjust transportation routes and schedules (L. Li, Negenborn, and De Schutter 2013). In order to solve the problem of integrating passenger and freight services on the same railway network and maximize the transportation efficiency and cost-effectiveness of the network, S. Li et al. (2023) imposed constraints on train, station capacity and freight delays, then calculated the minimum service and routing costs. Only in recent years has short-haul transport, especially intermodal transport within cities, begun to gain popularity (Zhu et al. 2023).

Passenger-Freight intermodal transport by bus is one of the most realistic forms in the cities and is most similar to the Modular Autonomous Vehicles (MAVs) studied in this thesis. Zeng and Qu (2022) built the network taking into account customer pick-up time windows, loading/unloading service duration, and the power supply needs of electric buses. To minimize operating costs, a Mixed Integer Linear Programming model is developed to meet the bus schedule for passenger travel needs, cargo delivery needs and charging requests. Machado, Pimentel, and Sousa (2023) further considered the uncertainty of cargo demand and establishes a model through demand scenarios in different time and space. The buses in the current study all need to be modified, with fixed passenger and cargo capacity, and there will still be problems with redundant passenger or cargo compartments in reality. The dynamic demand of passengers for direct access cannot be met by buses, and other modes of transportation are required for the last mile (Machado, Pimentel, and Sousa 2023). The small capacity of each unit and the coupling/decoupling characteristics of MAVs can effectively make up for the shortcomings of buses in urban intermodal transport (Shi and X. Li 2021).

3.2. Optimization methods related to MAU

In recent years, with the development of autonomous driving technology, the emergence of modular vehicles has opened up new directions for the exploration of public transportation networks. When establishing the optimal timetable for each bus route based on the minimum cost network flow model, Hassold and Ceder (2014) proposed to use multiple small vehicles to replace the original planned models. The results showed that this method can significantly reduce operating costs and improve service

quality. Modular autonomous vehicles (MAVs) can change the formation by changing the number of composed MAUs according to the time-varying passenger demand at different stations (Shi and X. Li 2021), so more research is currently focused on establishing a cost-controlled public transportation network that serves passengers with travel needs.

Liu, Qu, and X. Ma (2021) proposed that each MAU can freely visit customers outside the checkpoint, and deal with the first-mile and last-mile passenger pick-up problem by determining the best route and scheduling strategy. A Mixed Integer Linear Programming that not only focuses on vehicle operating costs, passenger waiting and in-vehicle costs, but also adds penalties for not responding to passenger needs is designed. To solve this problem, they used a customized dynamic programming with valid cuts in the first stage to effectively arrange the dynamic time and route of vehicles, and proposed an effective and fast heuristic method in the second stage to solve the dynamic allocation and path problem, and obtained a plan for real-time allocation and scheduling of vehicles in a region. Tang et al. (2024) considered that the areas between stations should be distinguished, and the service areas and routes of the day are selected according to the passengers' online reservations one day in advance, which will cause some passengers to walk to another nearby station that will be served. The MAU will separate from the MAV fleet at the starting point of the selected section and reconnect with the fleet at the end of these sections. They developed an Optimization model to determine the deviated sections and the number of MAUs to be separated, as well as the corresponding schedule, to minimize the waiting time of passengers and the total cost of operator operation. The results are obtained using the DICOPT solver and show that the number of door-to-door passengers would affect the total cost. The more such passengers there are, the higher the waiting time and walking time cost will be, while the in-vehicle time and vehicle operation cost will be reduced.

Xia, J. Ma, Sharif Azadeh, and W. Zhang (2023) focused on the uncertainty of passengers' time-dependent demand, and seeked the lowest-cost and best robust MAV time-varying and station-varying capacity allocation plan and the corresponding schedule. Due to the random nature of passenger arrival rate, the number of MAVs will change over time in a trip, and different MAVs will have different numbers of MAUs at the same station. A model of the Timetable and Dynamic Capacity Allocation (TT-DCA) problem is proposed and extended by the Data-Driven Distributionally Robust Optimization (DRO) method to consider the uncertainty of passengers' arrival time. The Integer L-shaped (IL) method can better solve this problem. The results show that the allocation strategy of time-varying and station-varying capacity can lead in both maximum and average vehicle congestion levels, indicating that the waste of space in the vehicle can be effectively reduced. According to Xia, J. Ma, and Sharif Azadeh (2024a), the passenger demand for direct access was added to the fixed route, and the MU at any station can be separated from the MAV to complete the DRT service, and after completing the service, it can continue to couple with the MAV that coincides with the time. The Mixed Integer Linear Programming model is established to generate a globally optimal co-mobility schedule and service route (minimizing the weighted sum of passenger and operating costs). By using a customized Adaptive Large Neighborhood Search (ALNS) algorithm combined with the GUROBI solver, it is found that the OT-FC-DRT strategy makes the number of operating vehicles and operating costs higher than the OT-FC strategy, but reduces the waiting time cost and average in-vehicle time of passengers. If the cost weight is adjusted to favor the operator, the DRT service may be sacrificed. Furthermore, the dynamic constraint of passengers transferring between different routes was added by Xia, J. Ma, and Sharif Azadeh (2024b), so that the model can realize the function of finding a scheduling scheme and schedule that utilizes fewer MUs while establishing multi-line circulation. Based on the Integer L-shaped (IL) method, the Rolling Horizon Framework (RH) has been proposed to address efficiency issues found in numerical experiments based on Beijing's bus network data. It works by dynamically and incrementally solving the problem to adapt to the real-time changes and uncertainties in demand. For the three modes of Fixed-capacity, Partially flexible-capacity and Completely flexible-capacity, the costs for passengers do not differ much in different cases, but the Completely flexible-capacity mode has huge advantages in terms of operating costs and the number of vehicles used. However, if more attention is paid to the interests of passengers, the operating costs will increase, but the proportion of transfers within the vehicle can be increased, making it more convenient for passengers to transfer.

The use of MAVs for passenger and freight transport is a relatively new concept. Due to the low demand of passengers during off-peak periods, there is more redundancy in vehicles, while the rise of e-shopping has increased the demand for sending and receiving goods. MAVs that can quickly con-

nect and detach are a promising solution for a co-modal system that integrates public transportation services and last-mile logistics (Lin, Nie, and Kawamura 2022). Hatzenbühler et al. (2023) proposed two independent dedicated MAUs for passengers and freight. They modeled the Modular Multi-purpose Pickup and Delivery Problem (MMP-PDP) by considering the travel time cost, travel distance cost, fleet size cost, and the cost of unserved requests. The transportation mode only involves demand response type. Under the condition of meeting the time window constraints, the corresponding group of MAVs are arranged to depart from the depot and complete a series of tasks of picking up and dropping off passengers and goods in the shortest path before returning to the depot. Experiments show that while the operating cost is reduced by 48%, the travel time is also much shorter than that of ordinary fleets. Lin and F. Zhang (2024) proposed a different concept, where the same MAU serves both passengers and cargo. The number and task allocation of MAUs can be adjusted at any station to minimize the waiting time cost of passengers and the operating cost of operators. A fixed route in both directions is established, but since the two directions share the module inventory at the same station, the planning of both directions will affect each other. In order to simplify the established Mixed Integer Programming problem, the time coordinate is shifted to focus on the time when each MAV departs from the first station. A two-stage heuristic algorithm is used to solve this problem. The first stage determines the number and the timetable of vehicles that need to be scheduled, and the second stage develops a high-quality lower bound to optimize vehicle grouping and freight allocation. They finally found that an increase in freight demand and an increase in the maximum allowed platoon length can further reduce the total cost.

The following Table 3.1 shows the current research status of Modular autonomous vehicles (MAVs) and the research direction of this thesis. At present, most of the research on MAVs with only passengers is exploring the service mode that combines Fixed-route transit (FRT) and Demand-responsive transit (DRT). Such an intermodal mode can better reduce the empty vehicle rate and improve passenger satisfaction. However, there are fewer studies on passenger-freight intermodal transport, and most of them are one of the two service forms of FRT (Lin and F. Zhang 2024) and DRT (Hatzenbühler et al. 2024). The combination of these two service forms has been found to be more effective in previous studies on only passengers, so it is necessary to explore the impact of the combination of these two service forms on the cost control of MAVs' passenger-freight i2ntermodal transport.

Pas: Passenger, Fre: Freight, FRT: Fixed-route transit, DRT: Demand-responsive transit, DP: Dynamic programming, IL: Integer L-shaped, ALNS: Adaptive large neighborhood search algorithm, RH: Rolling horizon framework

Publications	**Demand**	**Transport Mode**	**Flexibility of Vehicle Formation**	**Objective**	**Solution**
Liu, Qu, and X. Ma (2021)	Pas	FRT + DRT	Docked/undocked at each stop	Operating cost, Waiting time cost, Travel time, Penalty cost for unserved demand	DP + Heuristic
Xia, J. Ma, Sharif Azadeh, and W. Zhang (2023)	Pas	FRT	Docked/undocked at each stop	Operating cost, Waiting time cost	IL
Tang et al. (2024)	Pas	FRT + DRT	Docked/undocked at selected stops	Vehicle ownership, Operating cost, Maintenance cost, Waiting time cost, Travel time	Solver (DICOPT)
Xia, J. Ma, and Sharif Azadeh (2024a)	Pas	FRT + DRT	Docked/undocked at each stop	Operating cost, Waiting time cost, Travel time	ALNS + Solver (GUROBI)
Xia, J. Ma, and Sharif Azadeh (2024b)	Pas	FRT + DRT (network)	Docked/undocked at each stop	Operating cost, Waiting time cost	IL + RH
Hatzenbühler et al. (2023)	Pas + Fre	DRT	Docked/undocked at terminals only	Operating cost, Vehicle ownership, Travel time, Penalty cost for unserved demand	ALNSA + Solver (CPLEX)
Lin and F. Zhang (2024)	Pas + Fre	FRT	Docked/undocked at each stop	Operating cost (vehicles and stations), Waiting time cost, freight handling cost	Two stage Heuristic
this thesis	Pas + Fre	FRT + DRT (network)	Docked/undocked at each stop	Operating cost	ALNS + Solver (GUROBI)

Table 3.1: Comparison of Various MAVs Studies

4

Problem description

This study considers the transportation services of multiple bus lines in a region during operation. The fixed lines have starting and ending points, set as depots, denoted by $d \in D$. The remaining stops are denoted by $s \in S$. There are transfer stops between different lines, intersections of two lines. These stations are represented by different numbers s on different lines but are spatially the same. Independent of the lines, the freight pickup and delivery stops are also denoted by $s \in S$, but these stops can change daily for providing door-to-door service. Modular Autonomous Vehicles (MAVs) pick up and drop off passengers only on fixed routes, while for freight, they can automatically move within the region to respond to demand at pickup and delivery stops.

In the bus system, the distribution of passengers typically exhibits dynamic characteristics at different times and different stations. During peak hours, the number of passengers is usually significantly higher than during off-peak hours, which results in more redundancy of MAUs during off-peak periods. Therefore, utilizing the remaining vehicle capacity to transport goods during off-peak periods, while meeting passenger transportation demands, is a more effective method to improve utilization. Adopting a flexible grouping mode that combines passenger and freight transport can significantly reduce the operational cost losses caused by the idleness of MAUs while ensuring service levels. Each MAU can couple/decouple at various stations and can be used for both passenger and freight transport. Since the automatic robot can assist the freight to move between different connected MAUs, as shown in Figure 4.1, and passengers can also freely shuttle between MAUs, as shown in Figure 4.2, this study assumes that both passengers and freight can reach the correct MAU before transferring, so there will be no penalty for transfers included in the cost. To maximize the utilization of vehicle space, passengers and freights can coexist in the same MAU when operating on fixed routes. However, if an MAU needs to go to a pick-up and delivery stop, it cannot have passengers on board. This transport mode is shown in Figure 4.3. Considering that MAUs can have different entrance and exit stations and time windows, each MAU is denoted by $k \in K$, and has a capacity limit Q.

This thesis focuses on the MAV Routing Problem in a multi-depot system with intersecting MAV bus lines, employing flexible grouping and rerouting modes to meet both passenger and freight transportation demands. The objective of the mathematical model is to minimize the operational costs of all used MAUs. It is assumed that the schedules of MAVs, the number of MAUs, node locations, and the quantities of passenger/freight transportation demands are known. The bus timetable is known (FRT), but when a MAU is free, it can start at any time to perform the task of transporting freight and assist in completing some passenger demands (DRT). Additionally, the paths between each station are associated with pre-determined operational costs (proportional to travel time), and it is assumed that the power consumption is the same whether there are passengers or freight in the MAU. The main reason why the cost of MAU is linked to travel time is that this study mainly considers the operating costs that operators need to pay and travel time affects energy consumption and battery efficiency. The longer the electric bus runs, the higher the energy consumption, resulting in higher operating costs. And for the same driving distance, in order to compare the difference in cost between peak hours and off-peak hours, it is more accurate to use time as the objective. The purpose of the study is to determine the

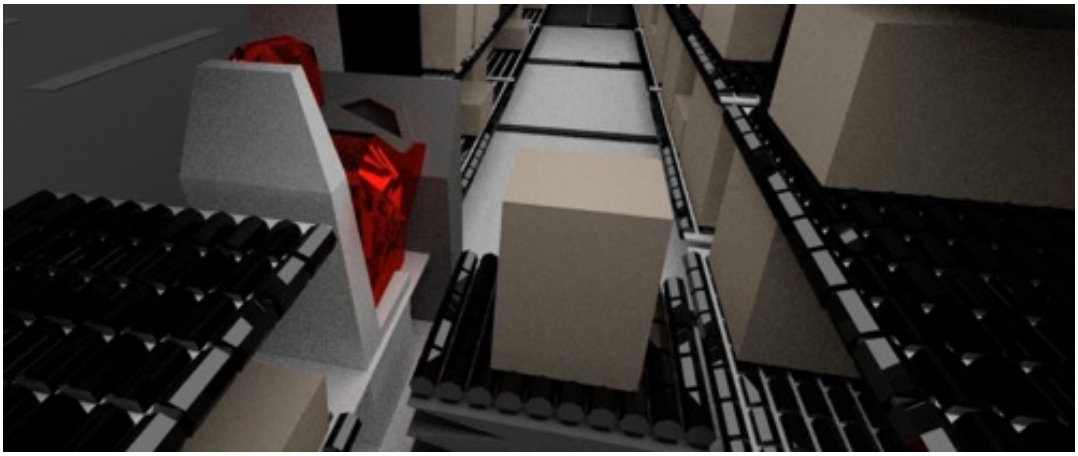

Figure 4.1: Freight movement between MAUs

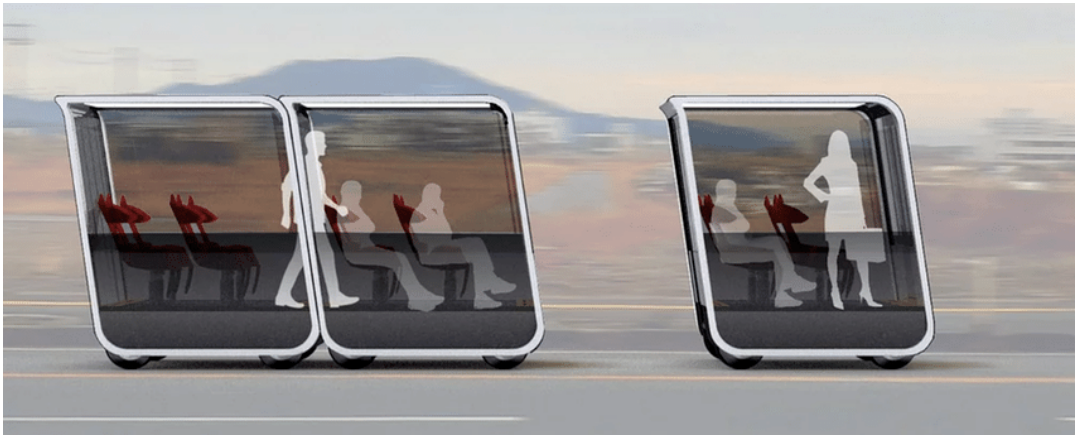

Figure 4.2: Passenger movement between MAUs

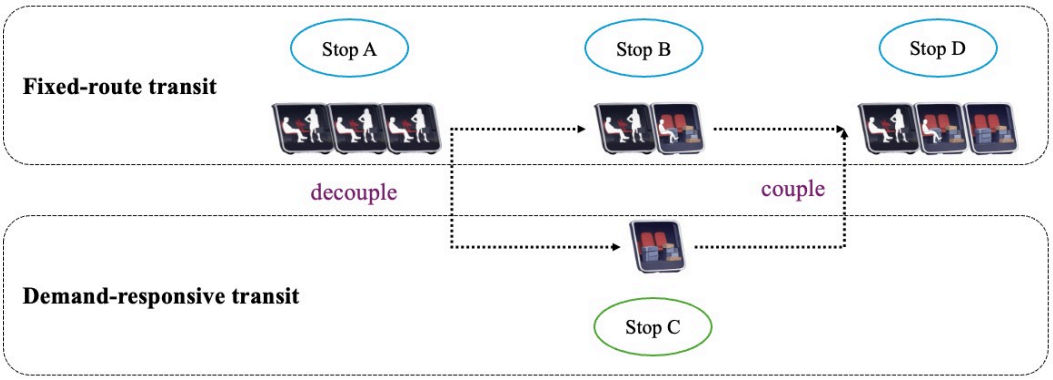

Figure 4.3: MAV transport mode

space-time trajectories of all operating MAUs. To describe the Space-Time movements of vehicles, passengers, and freight in the bus system, the Space-Time network representation method is introduced in the modeling process.

4.1. Space-Time network

The Space-Time network can clearly capture the working status and movement trajectory of each MAU at different timestamps. In the network, MAUs with coinciding Space-Time states automatically form

a column of MAVs. To construct the Space-Time network, the time range is discretized into a set of timestamps according to a time granularity, denoted as $T = \{t_0, t_1, ..., t_T\}$. The timestamp is divided into 1-minute intervals, so there are 1440 timestamps in one day. Depots and Stops are expanded into a series of Space-Time vertices at discrete timestamps. A Space-Time vertex can be represented in a unified form as (d, t) or (s, t), indicating the location of an MAU at timestamp t. The set N^R represents all Space-Time vertices. The movement of MAUs can be represented by directed arcs between Space-Time vertices. The set of Space-Time arcs is denoted by A^R, where each Space-Time arc is represented as (d, t, s, t') or (s, t, s', t') or (s, t, d, t'), describing the movement process of an MAU from the starting point to the destination. t represents the timestamp at station i, and $t' = t + t_a$, where t_a is the travel time from station i to station j. t_a is a three-dimensional matrix, as traffic conditions vary at different timestamps, and travel times differ for the two directions between stops. Additionally, travel times between the same two stations can vary at different timestamps. The travel time is expressed as follows:

$$t_a = \left(\begin{array}{c|ccccc||c|ccccc} t_0 & s_0 & s_1 & \cdots & d_n & & t_0 & s_n & s_{n-1} & \cdots & d_0 \\ d_0 & t_{0a_0} & & & & & d_n & t_{0a_{n+1}} & & & \\ s_0 & & t_{0a_1} & & & & s_n & & t_{0a_{n+2}} & & \\ \vdots & & & \ddots & & & \vdots & & & \ddots & \\ s_n & & & & t_{0a_n} & & s_0 & & & & t_{0a_{2n}} \end{array}\right) \times T \tag{4.1}$$

The travel time between stations, for example, from timestamp t_0, the travel time from stop s_n to depot d_n is denoted as t_{0a_n}. And there will be T (the number of all timestamps) matrices of this type.

A detailed description of nodes, arcs and paths is provided. Nodes are divided into stations and depots. Arcs are divided into travel arcs, delivery arcs, transfer arcs, and waiting arcs. Paths are formed by connecting these arcs in the order of stations and timestamps.

(1) Nodes

Depot Node: The set of these nodes is denoted as N_d, indexed by $n_d : (d, t)$. These nodes represent the starting and ending stations of all routes. For example, (d_1, t_0) and (d_2, t_0) represent the starting and ending points of Line 1 at the first timestamp. Each route has a pair of corresponding depots as starting and ending points at each timestamp.

Stop Node: The set of these nodes is denoted as N_s, indexed by $n_s : (s, t)$. These nodes represent all stops except the starting and ending stations of fixed routes. For example, (s_1, t_0) is the stop node for the first stop of Line 1 at the first timestamp. Each stop on each route at each timestamp has a unique identifier. Delivery and pick-up nodes outside the fixed routes start after the numerical identifiers of the stops on all routes. Some stops on fixed routes are transfer stops, meaning that although the station numbers are different on different routes, their physical locations are the same.

(2) Arcs

Traveling Arc: The set of these arcs is denoted as A_t, describing the movement of MAUs between stations on a fixed route, indexed by (i, t, j, t'). i and j are stops only on the fixed route or depots.

$$A_t \in \{(i, t, j, t') \,|i = (d_n, s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}),\ j = \left(s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}, \ldots, d_n\right), t' = t + t_a\}$$

Delivery Arc: The set of these arcs is denoted as A_d, describing the movement of MUs between delivery nodes outside the fixed routes or connecting to stops or depots on the fixed routes, indexed by (i, t, j, t'). The attributes are similar to those of travel arcs, but the connected stations are different.

$$A_d \in \{(i, t, j, t') \,|i = (d_n, s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}),\ j = \left(s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}, \ldots, d_n\right), t' = t + t_a\}$$

Transfer Arc: The set of these arcs is denoted as A_r, describing the movement of MAUs between stops on different routes that share the same physical location, indexed by (i, t, j, t'). i and j are predefined

stops, and MAUs can move freely between transfer nodes. $t' = t + \Delta$, where Δ is the predetermined transfer time required for MAUs.

$$A_r \in \{(i, t, j, t') \,|i = (s_n, s_{n+m}) \,,\ j = (s_{n+m}, s_n) \,, t' = t + \Delta\}$$

Waiting Arc: The set of these arcs is denoted as A_w, describing the state where MAUs remain stationary, indexed by (i, t, i, t'). i represents predefined stations where MAUs can wait, including depots, transfer stops, and all delivery stops outside the fixed routes. $t' = t + \Omega$, where Ω is the predetermined waiting time.

$$A_w \in \{(i, t, i, t') \,|i = (d_n, s_n) \,, t' = t + \Omega\}$$

(3) Paths

Each path is composed of a certain number of different types of arcs, and each path has two directions: forward and backward. The starting point of a path should begin at a depot and end at a depot. Additionally, the second set of stations and timestamps of the previous arc must match the first set of stations and timestamps of the subsequent arc. Only when both conditions are satisfied can a feasible path be formed. And since passengers can only be transported on fixed routes, all paths are further divided into passenger paths and freight paths. Passenger paths include all depots and stops on a fixed bus line, while freight paths additionally include all delivery and pick-up stops.

4.2. Demand

This study assumes that daily passenger and freight demand are known in advance. Expressing demand as the quantity on each arc is due to the difficulty in predicting precise transportation needs for specific passengers or freight, while arc-based demand, derived from statistical data, is more reliable and easier to obtain. Additionally, although the demand model does not capture specific passenger or freight boarding and alighting behaviors but represents them in groups (number of passengers or freight arriving at a station within a certain period of time), this approach is more suitable for scenarios with substantial transportation demand within a region.

For passenger demand, since the bus lines' schedules are known, demand is concentrated before the bus is expected to arrive. Therefore, from timestamp t to timestamp t', if there are n passengers with transportation demand from stop s to stop s', it will be represented as $e_i = \{(s, t, s', t') : n\}$. For freight demand, due to its lower time sensitivity, it only needs to be transported within a certain time window. If there are n pieces of freight that need to be sent out from stop s between timestamp (t, t') and arrive at stop s' between timestamp (t'', t'''), it will be represented as $f_i = \{(s, (t, t'), s', (t'', t''')) : n\}$. Due to the greater time flexibility of freight, this will enable more efficient scheduling strategies for MAU across different time periods in a day.

4.3. Problem illustrations

To clearly illustrate the trajectories of MAUs in this study for transporting passengers and freight within a Space-Time network, as shown in Figure 4.4, a group of MAUs conducted a trip that simultaneously accommodated Fixed-route transit (FRT) and Demand-responsive transit (DRT), demonstrating the flexibility of MAUs in a multi-route passenger-freight integrated transport system. Additionally, Figure 4.5 supplements the location and load status of each MAU at every timestamp. For the location status, for example, d_0 indicates that the MAU is stationed at depot d_0 at this timestamp, while s_1, s_2 indicates that the MAU is traveling between stops s_1 and s_2 at this timestamp.

Three assembled MAUs, numbered k_1, k_2, and k_3, form a MAV, and depart simultaneously from the depot d_0 on Line 1, starting at node (d_0, t_0). At the starting point, k_1 carries both passengers and freight, while k_2 and k_3 transport only passengers. After traveling for one unit of timestamp via the travel arc (d_0, t_0, s_0, t_1), they arrive at the next stop. At this point, k_2 and k_3 have sufficient capacity to handle the demand from s_1 to s_2, allowing k_1 to fulfill a demand-responsive transit within the time window. The passengers in k_1's carriage transfer to k_2 or k_3, enabling k_1 to detach and perform a delivery task. At

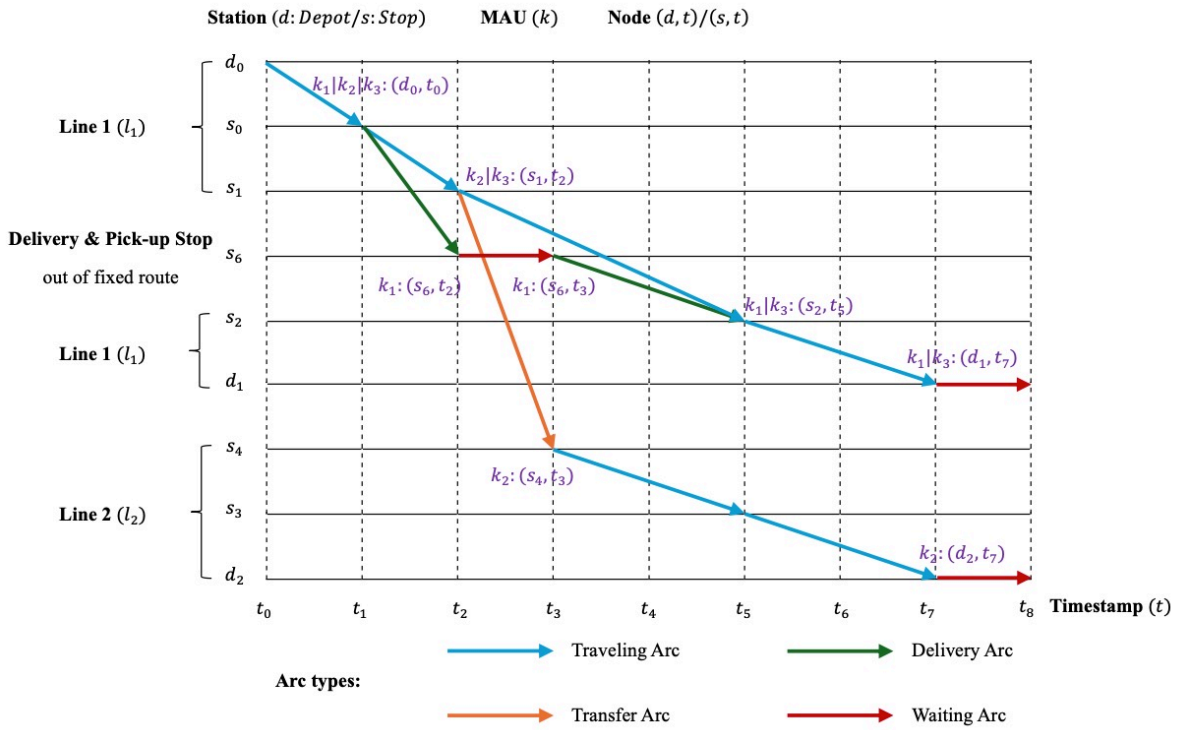

Figure 4.4: MAU's transport trajectory on the Space-Time Network

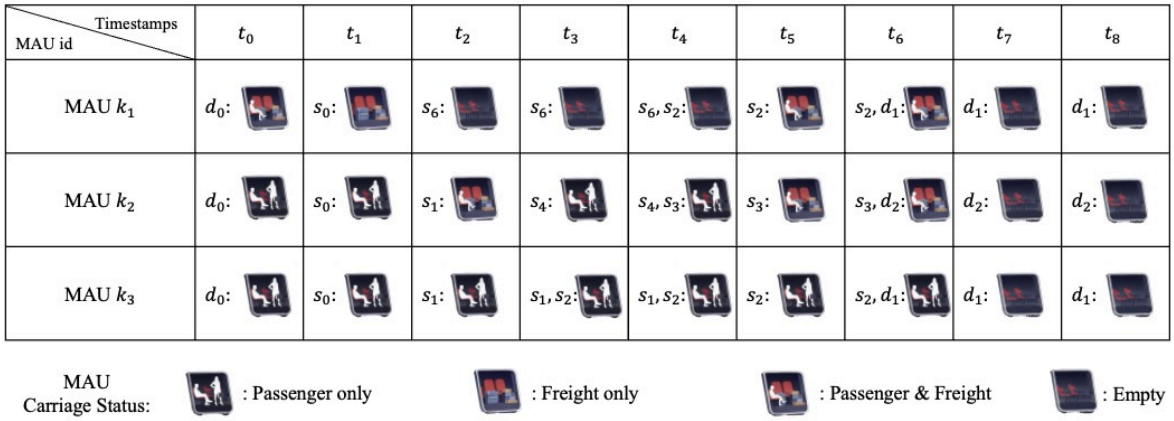

Figure 4.5: MAU's Position and Loading/Unloading status at each timestamp

timestamp t_2, k_1 arrives at the delivery and pick-up stop s_6 and unloads all freight in its carriage, while k_2 and k_3 arrive at s_1. Since k_3 has sufficient capacity to meet the demand on (s_1, t_2, s_2, t_5), the idle k_2 can transfer to Line 2 via the transfer arc (s_1, t_2, s_4, t_3) to continue service, transporting passengers from the geographical locations s_1 and s_4 to s_3 via (s_4, t_3, s_3, t_5). The passenger and freight demand on (s_2, t_5, d_1, t_7) exceeds k_3's capacity. After waiting for one timestamp via the waiting arc (s_6, t_2, s_6, t_3), the idle k_1 travels to s_2 and reassembles with k_3 into a single MAU at t_5. Ultimately, k_1 and k_3 arrive at the terminal d_1 on Line 1 at t_7, while k_2 arrives at the terminal on Line 2 at t_7.

In this example, although all three MAUs depart from node (d_0, t_0) on Line 1, they take three different paths to complete the transportation tasks. Through dynamic formation, line switching, and spatiotemporal coordination, the MAUs significantly enhance the operational efficiency and adaptability of passenger and freight co-transportation in modular transit systems.

5

Mathematical formulations

This thesis proposes a mathematical model for the MAU Routing Problem of Multi-functional Autonomous Units (MAUs) in a passenger-freight Space-Time Network, aiming to minimize total transportation costs while satisfying passenger and freight demand constraints. The problem is based on a network structure comprising depots, stops, and timestamps, considering MAU path selection, passenger and freight transport capacity constraints, and restrictions on MAU formations on the same arc. The model determines the path allocation for utilized MAUs, the fulfillment of passenger and freight demands, and the balance of MAUs entering and leaving depots. The following mathematical model is formulated as a Mixed-Integer Quadratically Constrained Programming (MIQCP)problem, integrating the objective function and constraints to ensure efficient optimization of transportation plans in complex networks.

The problem involves Multi-Depot, Space-Time Network, MAUs, and passenger-freight intermodal transportation. Since the Path-based Model pre-generates feasible paths p, it offers greater flexibility and conciseness compared to other models, making it the preferred choice for addressing this problem. Firstly, because the network incorporates both time and space dimensions, the path-based model eliminates the need for additional constraints to ensure continuity in space and time, thereby simplifying optimization decisions. At the same time, path-based decision-making aligns better with the formation flexibility of MAVs. By selecting paths, MAUs avoid real-time computation of all possible arc combinations, reducing complexity. Furthermore, since passenger transport operates only on fixed routes while freight is picked up and delivered at all nodes, capacity allocation becomes more intuitive. This approach allows for a clear definition of whether a path is carrying passengers or freight.

In the following sections, all sets, parameters and decision variables are introduced in detail and the complete mathematical model is presented.

5.1. Notations

All sets establish the foundation for entities and their relationships within the MAU transportation system, creating a structured representation for indexing vehicles K, paths P, demands $E\&F$, and physical infrastructure. Parameter settings quantify the system's operational characteristics and constraints, including demand volumes γ_e&λ_f, vehicle capacity Q, and more. By adjusting these parameters, different scenarios can be simulated to observe the impact of changes in demand or capacity on decision-making. The core decision variable $y_{e,a}^k$, which the model seeks to optimize, determines the allocation of vehicles on paths and directly influences costs. The auxiliary variables $y_{e,a}^k$ and $z_{f,a}^k$ play a critical role in tracking the passenger/freight loading status of vehicles and the fulfillment of demand quantities on each arc. Table 5.1 shows the notations and explanations of the sets, parameters, and decision variables used in the model

Table 5.1: Notation and Description for the Model

Notation	Description
Sets	
D	Set of depots, $D = \{1, 2, ..., \|D\|\}$, indexed by d
S	Set of stops, $S = \{1, 2, ..., \|S\|\}$, indexed by s
I	Set of stations, $I = \{1, 2, ..., \|I\|\}$, indexed by i, j
T	Set of timestamps, $T = \{1, 2, ..., \|T\|\}$, indexed by t, t
N	Set of nodes, $N = \{(1, 1), (2, 2), ..., \|I, T\|\}$, indexed by n
A	Set of arcs, $A = \{(1, 1, 2, 2), (2, 2, 2, 3), ..., \|I, T, J, T'\|\}$, indexed by a
P	Set of paths of MU k, $P = \{(at_1, ..., \|A_t\|, \|A_d\|, \|A_r\|, \|A_w\|), ...\}$, indexed by p
K	Set of MAUs, $K = \{1, 2, ..., \|K\|\}$, indexed by k
E	Set of sectional passenger demand, $E = \{1, 2, ..., \|E\|\}$, indexed by e
F	Set of freight group demand, $F = \{1, 2, ..., \|F\|\}$, indexed by f
Parameters	
c_p	Costs of path p
Q	Maximum capacity of each MAU k
G	Maximum number of MAU formations on the same travel arc or delivery arc
ρ	Passenger and freight capacity occupancy ratio
γ_e	Total volume of sectional passenger demand e
λ_f	Total volume of freight group demand f
$B^-_{d,p}$	Number of MAUs leaving depot d in path p
$B^+_{d,p}$	Number of MAUs entering depot d in path p
θ^p_a	Binary, if arc a is in the path p, the value is 1, otherwise 0
Decision Variables	
x^k_p	Binary, if MAU k passes through the path p, the value is 1, otherwise 0
$y^k_{e,a}$	Integer, number of sectional passenger demand e carried by MAU k on arc a
$z^k_{f,a}$	Integer, number of freight group demand f carried by MAU k on arc a

5.2. Objective and Constraints

$$\min \quad \sum_{k\in K}\sum_{p\in P} c_p x_p^k \tag{5.1a}$$

$$\text{s.t.} \quad \sum_{p\in P} x_p^k \le 1 \quad \forall k \in K \tag{5.1b}$$

$$y_{e,a}^k \le \gamma_e \sum_{p\in P} \theta_a^p x_p^k \quad \forall k \in K, e \in E, a \in A \tag{5.1c}$$

$$z_{f,a}^k \le \lambda_f \sum_{p\in P} \theta_a^p x_p^k \quad \forall k \in K, f \in F, a \in A \tag{5.1d}$$

$$\sum_{a\in A}\sum_{k\in K} y_{e,a}^k = \gamma_e \quad \forall e \in E \tag{5.1e}$$

$$\sum_{a\in A}\sum_{k\in K} z_{f,a}^k = \lambda_f \quad \forall f \in F \tag{5.1f}$$

$$\sum_{p\in P}\sum_{e\in E} \theta_a^p y_{e,a}^k x_p^k + \frac{1}{\rho}\sum_{p\in P}\sum_{f\in F} \theta_a^p z_{f,a}^k x_p^k \le Q \quad \forall k \in K, a \in A \tag{5.1g}$$

$$\sum_{k\in K}\sum_{p\in P} \theta_a^p x_p^k \le G \quad \forall a \in A \tag{5.1h}$$

$$x_p^k \in \{0,1\} \quad \forall k \in K, p \in P \tag{5.1i}$$

$$y_{e,a}^k \in [0, \gamma_e] \quad \forall k \in K, e \in E, a \in A \tag{5.1j}$$

$$z_{f,a}^k \in [0, \lambda_f] \quad \forall k \in K, f \in F, a \in A \tag{5.1k}$$

Objective Function: The objective (5.1a) is to minimize the cost of all paths that MAU k passes through. Since the cost is directly linked to the length of travel time, this means that the solution which this study is looking for is to assign each vehicle the lowest total path travel time.

Path Assignment Constraint: Constraint (5.1b) ensures that each MU k select at most one path p.

Demand Capacity Constraints: Constraint (5.1c) ensures that for each MU k, the volume of sectional passenger demand e transported does not exceed the total amount of that demand γ_e, and that transport is allowed only when the path p selected by the MAU k supports the sectional passenger demand. Constraint (5.1d) ensures that for each MU k, the volume of freight group demand f transported does not exceed the total amount of that demand λ_f, and that transport is allowed only when the path p selected by the MAU k supports the freight group demand.

Demand Satisfaction Constraints: Constraint (5.1e) ensures that all sectional passenger demands γ_d are met. Constraint (5.1f) ensures that all freight group demands λ_f are met.

Vehicle Capacity Constraint: Constraint (5.1g) ensures each MAU k on the path p, its total load (passengers and freight) shall not exceed the capacity Q. At the same time, ρ is added to flexibly adjust the different capacity occupancy rules between passengers and freight.

Vehicle Grouping Constraint: Constraint (5.1h) ensures that the number of MAU formation on each arc does not exceed G. During peak hours, the MAU can be controlled to prioritize passenger demand rather than freight demand.

The optimization model comprehensively considers multiple key constraints to ensure the accuracy of path allocation, providing a feasible verification framework for the subsequent design of the ALNS algorithm. Path allocation, demand capacity, demand satisfaction, flow conservation, and vehicle capacity and grouping constraints collectively form the feasible region boundaries of the solution.

6

Solution methods

When selecting a computational method suitable for this MAU Routing Problem, it is considered that the model needs to simultaneously optimize the routing of each MAU, platoon formation, the integration of passenger and freight, and fleet size. This introduces a large number of binary decision variables (x_p^k) for assigning and integer variables ($y_{e,a}^k$ and $z_{f,a}^k$) for demand allocation, along with nonlinear quadratic constraints such as capacity limits, which increase the complexity of the solution process. For small-scale instances, exact solvers like GUROBI can find optimal solutions within an acceptable time frame. However, for medium or large-scale instances, exact methods relying on branch-and-bound algorithms face exponential growth in solution time as the number of variables increases for NP-hard problems, potentially leading to memory shortages. In practical operations, since transportation planning requires near-real-time rapid decision-making, the computational speed of exact solvers is insufficient, whereas heuristic methods can provide near-optimal solutions with minimal gaps in a shorter time.

Compared to other heuristics, the Adaptive Large Neighborhood Search (ALNS) algorithm can balance diversification and intensification, avoiding the risk of getting trapped in local optima while also maintaining convergence speed. The adaptive mechanism of ALNS allows for dynamic weight adjustments based on operator performance and enables the destruction and repair of large solution structures, enhancing search efficiency and solution diversity. Furthermore, the ALNS algorithm can be adjusted to account for modular formation characteristics, making it particularly suitable for scenarios where passenger and freight are integrated within the same MAU.

Based on these observations, this section focuses on designing a hybrid Adaptive Large Neighborhood Search (ALNS) algorithm at the arc and path levels to find high-quality solutions within acceptable computational time. Two initial solution generation methods are compared. First, a small number of feasible paths are generated based on demand, and then GUROBI or heuristic methods are used to obtain initial solutions. Building on this, ALNS is employed to explore feasible solutions that meet demand with lower costs. The overall process of the algorithm is shown in the Figure 6.1:

For this MAV Routing Problem, the solution of the ALNS algorithm must pass the following four feasibility checks:

1. All passenger and freight demands are satisfied without exceeding the capacity of each MAU;
2. The carriage of each MAU can be in states with only passengers, only freight, or both passengers and freight simultaneously;
3. After completing the transportation demand task on an arc, the MAU's capacity must be released, meaning that a single MAU should be able to fulfill multiple demands along a single path;
4. The number of MAU groups on each arc cannot exceed the limit.

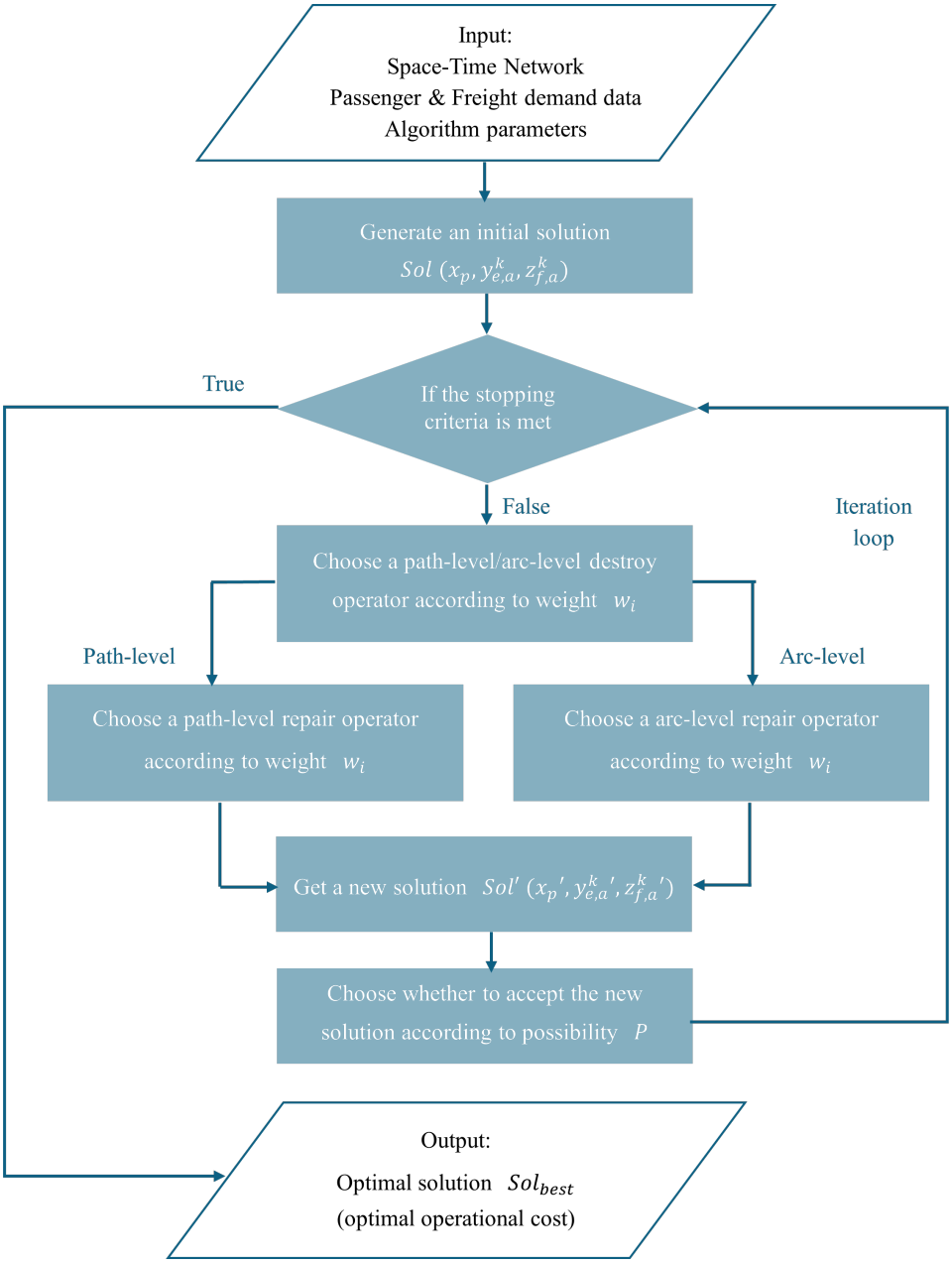

Figure 6.1: Overall flowchart of algorithm

6.1. Adaptive Large Neighborhood Search algorithm

Shaw (1997) first proposed a Local Search algorithm that employs greedy local search and utilizes a larger neighborhood to avoid local minima. Ropke and Pisinger (2006) extended this approach, improving it into the Adaptive Large Neighborhood Search (ALNS) algorithm. Traditional local search techniques typically explore only a limited subset of solutions, making only small modifications to the current solution and easily getting trapped in local optima. ALNS, however, can remove a significant portion of the solution and reconstruct it in a single iteration, while also allowing the use of multiple destruction and repair methods within the same search process, providing greater flexibility in finding optimal solutions.

ALNS consists of three main components: an initial solution, destruction/repair operations, and an acceptance criterion. The ALNS algorithm starts with an initial solution, which serves as the starting point for the iterative search process. In each iteration, the algorithm performs a series of operations

aimed at improving the current solution. These operations include destruction (removing a portion of the current solution using a specified removal operator) and repair (reinserting the removed portion using an insertion operator). The solution obtained after each iteration is called the incumbent solution and is evaluated through the acceptance criterion to determine whether it is accepted. Throughout the iterative process, the algorithm continuously updates the current solution based on the results of the destruction and repair operations, enabling ALNS to explore different regions of the solution space to find better solutions.

The score updating procedure tracks the performance of each heuristic algorithm, guiding the search process toward the most promising regions of the solution space. In each iteration, removal and insertion heuristic algorithms are applied to the current solution, and the scores of these operators are continuously updated. If a heuristic algorithm finds a new global best solution or discovers an unvisited solution that is accepted by the acceptance criterion, its score increases. However, if it performs poorly, failing to improve the solution or explore new regions, its score may decrease or remain unchanged. A higher score indicates that the heuristic algorithm is more successful in finding good solutions. This dynamic scoring mechanism ensures that the algorithm prioritizes recently high-performing heuristics, thereby improving search efficiency and the quality of solutions.

The new solution generated by modifying the current solution through the application of removal and insertion heuristic algorithms is referred to as the proposed solution. Whether to accept or reject this solution is determined probabilistically, inspired by the Simulated Annealing (SA) algorithm, based on the difference in the objective function values between the current solution Sol and the proposed solution Sol', as well as the current temperature parameter T. Table 6.1 illustrates the framework of the ALNS algorithm in this study.

Input: initial solution $Sol\ (x_p^k, y_{e,a}^k, z_{f,a}^k)$

While $i < \max Iteration$ **do**:

1 Select a destroy operator according to the probabilistic score

2 Select an insertion operator according to the probabilistic score

3 Generate a new solution Sol'

4 If Sol' is accepted then

  $Sol = Sol'$

  End

5 If $f(Sol) < f(Sol')$ then

  $Sol_{best} = Sol$

  End

6 Update the scores of the operators

7 Update the temperature & acceptance probability

8 $i = i + 1$

Output: Best solution Sol_{best}

End

Table 6.1: Framework of the ALNS algorithm

6.2. Acceptance criterion

The ALNS algorithm uses Simulated Annealing (SA) as the acceptance rule. The main idea of SA in ALNS is not only to accept improved solutions, but also to provide opportunities for accepting worse solutions. The SA process starts with a high initial temperature, allowing the acceptance of worse solutions to explore the solution space. As the process continues, the temperature gradually decreases, tending towards accepting improved solutions. According to Metropolis guidelines, if the objective function value of the new solution $f(Sol)'$ is better than the current solution $f(Sol)$, the new solution is accepted. Otherwise, a random number $rand$ between 0 and 1 is generated, and the new solution is

accepted with probability $P = e^{-\frac{f(Sol')-f(Sol)}{T}} > rand$. This probabilistic mechanism allows the algorithm to jump out of local optima in the early stage and gradually converge to high-quality solutions in the later stage. After each iteration, the temperature T is updated using the expression $T \cdot \psi$, where ψ is the cooling rate. In this study, the initial temperature T_0 is set to 100, and the cooling rate ψ is 0.995. The principles used by SA in the ALNS algorithm are shown in Table 6.2.

Input: initial solution $x_p^k, y_{e,a}^k, z_{f,a}^k$, initial temperature T, cooling rate ψ
While $i < \max Iteration$ **do**:
1 Generate a new solution Sol' after destroy and insertion processes
2 If $P = e^{-\frac{f(Sol')-f(Sol)}{T}} > rand(0,1)$ then
$Sol = Sol'$
End
3 If $f(Sol) < f(Sol')$ then
$Sol_{best} = Sol$
End
4 Update T
5 $i = i + 1$
Output: Best solution Sol_{best}
End

Table 6.2: Simulated Annealing in ALNS algorithm

6.3. Generation of initial solution

In order to shorten the time for generating the initial solution, both methods use a small number of feasible paths for decision making, which are called demand paths. All feasible paths are called valid paths, denoted as P_{valid}. Demand paths are a subset of valid paths, extracted from valid paths based on arcs with transportation demands, denoted as P_{demand}. This can effectively reduce the number of decision variables that need to be considered in the initialization method.

6.3.1. GUROBI solver method

Although the GUROBI solver performs poorly with large-scale data, it can quickly and accurately obtain optimal decisions for small-scale data, providing a better starting point for ALNS compared to random generation or simple heuristics. This allows ALNS to converge to a satisfactory solution with fewer iterations, significantly reducing overall runtime. Additionally, solutions derived from the mathematical model always pass feasibility checks, avoiding the difficulty of finding feasible initial solutions under complex constraints in a short time. This hybrid strategy of exact solver and heuristic algorithms fully leverages their complementary strengths: GUROBI's mathematical optimization capability quickly identifies a high-quality starting point, while ALNS's neighborhood search ability further explores and improves the solution space.

6.3.2. Greedy heuristic method

This method maximizes the coverage capability of MAUs through iterative allocation to satisfy unassigned demand on paths. Priority is given to selecting paths with high coverage-to-cost ratios based on demand, where the cost can be balanced using this formula to obtain paths that satisfy the remaining transportation demand requirements:

$$s_p = \frac{1}{\sqrt{c_p}} \left(\sum_{e \in E} u_e \sum_{a \in P} \alpha_e^a + \frac{1}{\rho} \sum_{f \in F} u_f \sum_{a \in P} \beta_f^a \right) \tag{6.1}$$

where $u_e = \gamma_e - \sum_{a\in A}\sum_{k\in K} y^k_{e,a}$ represents the unassigned passenger demand, α^a_e is a binary variable indicating whether this arc can transport this passenger demand, and $\sum_{e\in E} u_e \sum_{a\in P} \alpha^a_e$ represents the total unassigned passenger demand that the path can cover. Similarly, $u_f = \lambda_f - \sum_{a\in A}\sum_{k\in K} z^k_{f,a}$ represents the unassigned freight demand, β^a_f is a binary variable indicating whether this arc can transport this freight demand, and $\frac{1}{\rho}\sum_{f\in F} u_f \sum_{a\in P} \beta^a_f$ represents the total unassigned freight demand that path p can cover.

A three-stage process is adopted: First, vehicles are allocated to paths to satisfy passenger demand. Under the constraint of adhering to capacity limits, MAUs with available capacity can satisfy freight demand, as first satisfy:

$$y^k_{e,a} \le \min(u_e, Q - \sum_{e\in E} y^k_{e,a}) \quad \forall k \in K, a \in A \tag{6.2}$$

then satisfy:

$$z^k_{f,a} \le \min\left(u_f, Q - \sum_{e\in E} y^k_{e,a} - \frac{1}{\rho}\sum_{f\in F} z^k_{f,a}\right) \quad \forall k \in K, a \in A \tag{6.3}$$

THen, all MAUs assigned to the same arc are checked for grouping situations. All demands must be satisfied under mandatory constraints, as $u_e = 0$ and $u_f = 0$. However, exceeding the maximum fleet size for a single arc may occur. This happens because this constraint are not subject to high-priority requirements. In subsequent ALNS algorithms, this can be effectively addressed through disruption and insertion processes. This iterative startup method can find an initial solution in a very short time and can greatly accelerate the solution speed.

6.4. Destroy heuristics

This study categorizes destroy operators into two levels: path and arc. Path-level destruction removes entire paths, which can release large solution spaces and significantly reconstruct MAU path allocation, while Arc-level destruction preserves useful portions of paths and more precisely removes arcs that have no demand but occupy costs. Figure 6.2 shows the path-level destroy process, while Figure 6.3 shows the arc-level destroy process. Each type of operator has its own advantages and disadvantages, working together to maintain the depth of local optimization while providing the breadth of global search. This section explains the rationale and principles behind using these ALNS removal heuristic methods.

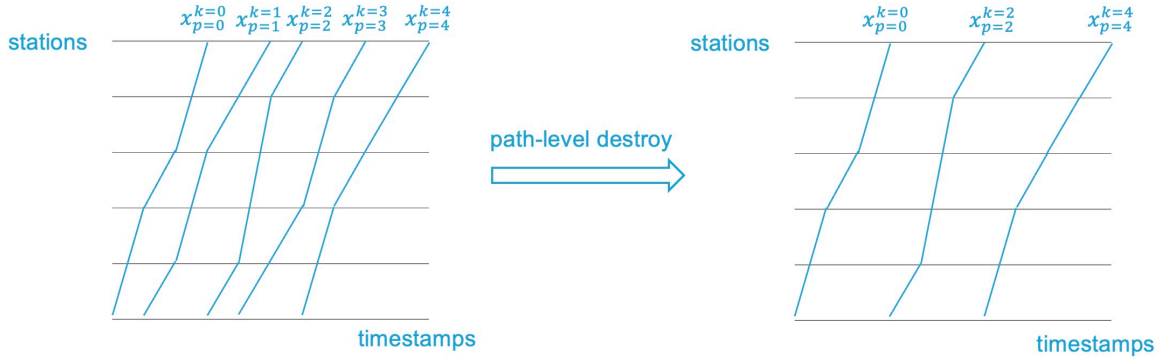

Figure 6.2: Path-level destroy process

6.4.1. Dynamic destroy rate

The destruction rate is a parameter that controls the proportion of vehicle paths removed from the solution in each iteration, denoted by ξ and constrained within $[\min\xi, \max\xi]$. This ensures that each destruction operation has a certain degree of disturbance, avoiding the algorithm being too conservative while preventing excessive destruction that would lead to complete randomization of the solution. When no better solution is found after a certain number of iterations, the destruction rate is increased using $\xi' = \min(\max\xi, \xi + 0.05)$ to encourage the algorithm to explore new solution spaces. When a better solution is found, the counter is reset to 0, and the destruction rate is decreased using

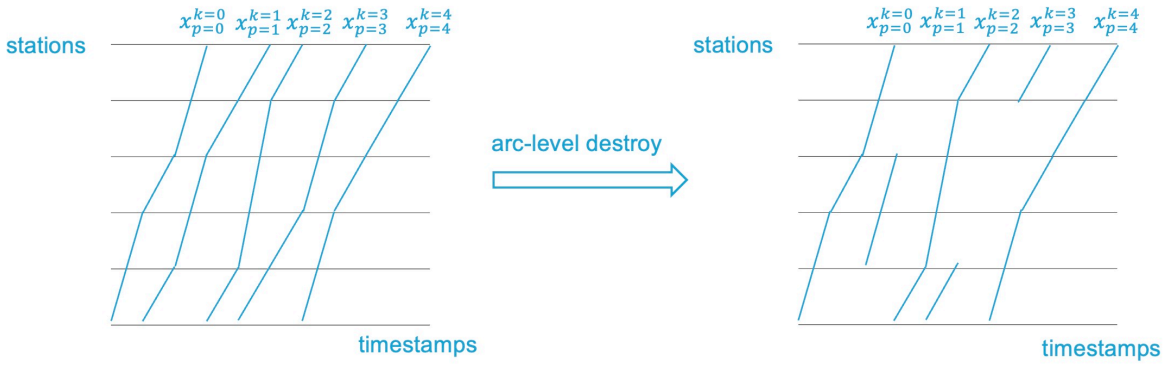

Figure 6.3: Arc-level destroy process

$\xi' = \max(\min \xi, \xi - 0.02)$ to make the algorithm more inclined to perform fine optimization within the neighborhood of the current solution. The dynamic change process is shown in Table 6.3. The target number of vehicle allocation paths to be removed for each operator is:

$$Sol_{target} = \max(1, \lfloor Sol_{active} \times \xi \rfloor) \tag{6.4}$$

where Sol_{active} is the set of vehicles that have already been allocated paths in the current solution. However, the actual number of removals is constrained to avoid completely destroying the solution by:

$$Sol_{remove} = \min(Sol_{target}, \lfloor Sol_{active} - 1 \rfloor) \tag{6.5}$$

Input: initial destroy rate ξ , destroy rate range $[\min \xi, \max \xi]$

While $i < \max Iteration$ **do**:

When solution Sol has no improvement:
  no improvement count =+ 1
  When no improvement count $\geq$ n
    $\xi' = \min(\max \xi, \xi + 0.05)$
  no improvement count = = 0

When solution Sol has an improvement:
    $\xi' = \max(\min \xi, \xi - 0.02)$
  no improvement count = = 0

End

Table 6.3: Dynamic destroy rate process

6.4.2. Path-level destroy operator

(1) Random Destroy

The core principle of Random Destroy is to disrupt the structure of the current solution by randomly selecting and removing a certain number of vehicle routes. Although it is the most basic destruction strategy, it is highly important. The primary reason for using this operator is its ability to provide essential diversity to the algorithm. When trapped in a local optimal solution, it can unbiasedly explore various corners of the solution space, effectively preventing premature convergence. Additionally, randomness helps the algorithm escape its current search trajectory, offering fresh reconstruction opportunities for the subsequent repair phase. This random perturbation mechanism is key to maintaining the algorithm's long-term search vitality and often unexpectedly discovers new regions containing high-quality solutions. The specific steps are shown in Table 6.4.

(2) Worst Cost Destroy

Input: Current Solution Sol
Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$
1 **Select** $|Sol_{remove}|$ paths *randomly*
2 **Remove**
Output: Partially Destroyed Solution Sol'
End

Table 6.4: Random Destroy Process

The design principle of Worst Cost Destroy is based on a key optimization intuition: the highest-cost routes often represent the least efficient resource allocations in the current solution, making their removal most likely to yield significant cost improvements. This operator is distinctly goal-oriented, targeting routes with suboptimal costs and providing the algorithm with a clear direction for improvement. Since a few high-cost routes typically contribute disproportionately to the total cost, removing them can create the greatest potential for improvement in the subsequent repair phase. Moreover, the presence of high-cost routes often suggests that the current route selections may be suboptimal, indicating the potential for alternative solutions with more efficient resource utilization. The specific steps are shown in Table 6.5.

Input: Current Solution Sol
Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$
1 **Sort** cost of each path of x_p^k
2 **Select** top $|Sol_{remove}|$ *highest-cost* paths
3 **Remove**
Output: Partially Destroyed Solution Sol'
End

Table 6.5: Worst Cost Destroy Process

(3) Demand-based Destroy

The design principle of Demand-based Destroy is rooted in the concept of optimizing transportation efficiency, which involves reconfiguring vehicle routes with low demand service efficiency, as identified by the key index 'efficiency = number of transportation demand / path cost'. During off-peak periods, there may be vehicle routes with relatively low passenger and cargo loads or low service demand density. Even if these routes have modest costs, low-efficiency vehicles often indicate resource waste, suggesting suboptimal route planning that results in insufficient vehicle loads. By removing these inefficient vehicles, the algorithm creates opportunities to reorganize and optimize demand allocation, forcing the repair phase to seek solutions that better utilize vehicle capacity and serve higher-value demands more effectively. The specific steps are shown in Table 6.6.

6.4.3. Arc-level destroy operator

(1) Arc Removal Destroy

The core of Arc Removal Destroy is to remove a certain number of non-demand arcs from an allocated path, and then select the segment containing the most demand from the disconnected segments as the new path. First, the paths with the most non-demand arcs are selected. For a path, the target number of non-demand arcs to be removed is controlled within the range $A_{target} = \text{random}\,(1, \xi \times A_{active})$, where A_{active} refers to the total number of arcs contained in this path, and the actual number of removals satisfies $A_{remove} = \min(A_{target}, A_{non-demand})$, where $A_{non-demand}$ refers to the number of all non-demand arcs on this path. After completing the removal steps, there will be multiple segments containing demand remaining, and only the segment containing the most demand is retained to avoid

Input: Current Solution Sol

Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$

Calculate efficiency for each path of x_p^k $(Efficiency(p_i) = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}])$

Select top $|Sol_{remove}|$ *lowest-efficiency* paths

Remove

Output: Partially Destroyed Solution Sol'

End

Table 6.6: Demand-based Destroy Process

the situation where arcs with demand have only one unique connectable non-demand arc, providing more exploration space in the repair phase. The specific steps are shown in Table 6.7.

Input: Current Solution Sol

Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$

Sort non-demand arcs contained in each path of x_p^k

Select top $|Sol_{remove}|$ *most non-demand arcs* paths

For each selected path:

  Select all non-demand arcs:

    Set $|A_{remove}| = min(A_{target}, A_{non-demand})$,
    where $A_{target} = random(1, \xi \times A_{active})$

    Select $|A_{remove}|$ *non-demand* arcs

    Remove

  Identify segments of the path containing demand:

    Select the segment with *most demand arcs*

      Append the segment as a new path into Sol'

Output: Partially Destroyed Solution Sol'

End

Table 6.7: Arc Removal Destroy Process

(2) Arc Search Destroy

Arc Search Destroy is a destruction operator based on transportation efficiency analysis of each arc. First, similar to Demand-based Destroy, the least efficient paths are found, but the entire path to which the MAU is assigned will not be removed. Then the transportation efficiency of each arc on the path to be destroyed is calculated according to the formula $Efficiency(a_i) = \frac{1}{c_{(a_i)}}[\gamma_{(a_i)} + \frac{1}{\rho}\lambda_{(a_i)}]$, where $\gamma_{(a_i)}$ is the number of passengers transported by arc a_i, $\lambda_{(a_i)}$ is the freight quantity, and $c_{(a_i)}$ is the cost required to pass through this arc. After sorting each arc segment, the ξ least efficient ones will be removed. Since non-demand arcs have a numerator of 0 in the efficiency calculation, most of them will be preferentially removed. If more than half of the arcs serve no demand, random selection will be performed. Through destroying empty arcs or low-value arcs with excessively high costs, this provides a better foundation for subsequent repair operators. The specific steps are shown in Table 6.8.

6.5. Insertion heuristics

To repair the removed vehicle paths or arc segments, ALNS requires insertion heuristic methods to regain new complete solutions. This study employs three path-level repair operators: Random Repair, Greedy Repair, and Utility Maximization Repair, along with three arc-level repair operators: Arc Inser-

Input: Current Solution Sol

Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$

Calculate efficiency for each path ($Efficiency(p_i) = \frac{1}{C_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$)

Select top $|Sol_{remove}|$ *lowest-efficiency* paths

For each selected path:

  Calculate efficiency for each arc ($Efficiency(a_i) = \frac{1}{C_{(a_i)}}[\gamma_{(a_i)} + \frac{1}{\rho}\lambda_{(a_i)}]$)

  Sort arcs by Efficiency

  Select $|A_{remove}|$ *lowest-efficiency* arcs

  Remove

  Identify segments of the path remaining:

    Append the segments as a new path into Sol'

Output: Partially Destroyed Solution Sol'

End

Table 6.8: Arc Search Destroy Process

tion Repair, Chain Repair, and Hybrid Repair. The destroy operators at path and arc levels are matched with corresponding repair operators at the same level to ensure repair quality. Figure 6.4 shows the path-level repair process, while Figure 6.5 shows the arc-level repair process. This section explains the reasons and principles behind using these ALNS insertion heuristic methods.

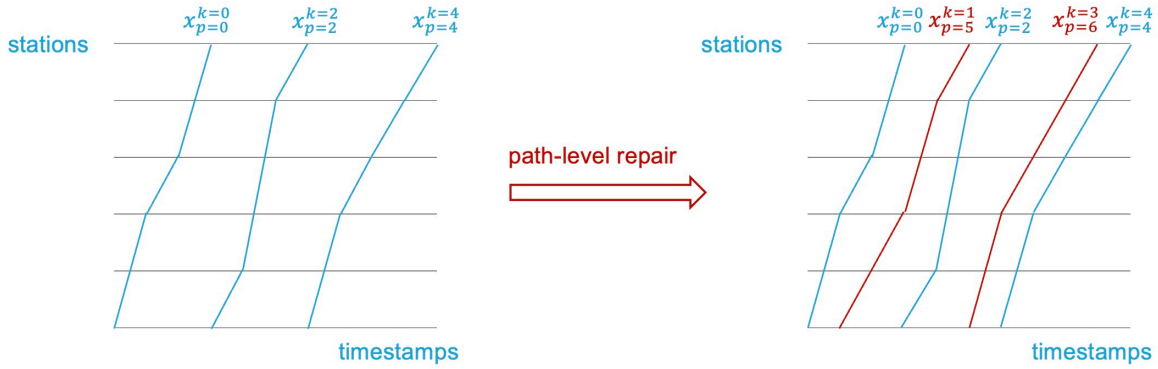

Figure 6.4: Path-level repair process

6.5.1. Candidate paths

To improve algorithm efficiency, after certain demands are removed by the destroy operator, the algorithm does not blindly search through all possible paths. Instead, it prioritizes paths that are highly related to the removed demands, known as candidate paths. During the repair process, candidate paths serve two main roles. First, the search space is significantly reduced, excluding many meaningless paths and thereby substantially accelerating the algorithm's runtime. Second, this study establishes a path relevance scoring mechanism, where paths capable of serving multiple removed demands receive higher scores, enhancing the quality of the repair.

6.5.2. Path-level repair operator

(1) Random Repair

The working principle of Random Repair in this study involves selecting the feasible candidate paths

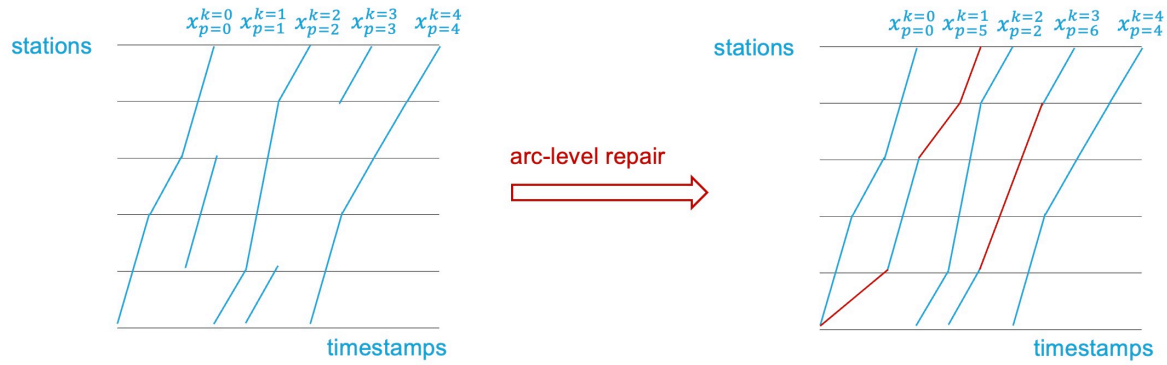

Figure 6.5: Arc-level repair process

which can cover the deleted demands and performing random selection under the premise of feasibility. The primary reason for using this operator is its ability to effectively counteract the search stagnation that may occur in the ALNS algorithm during prolonged iterations. When the solution structures remain similar over an extended period, random repair can break this regularity, maintaining the algorithm's creativity. The specific steps are shown in Table 6.9.

Input: Partially Destroyed Solution Sol'	
Generate candidate paths which can cover destroyed demand	
Iterate while u_e & $u_f \neq 0$:	
1	**Select** the path *randomly*
2	**Insert**
Evaluate feasibility	
Output: Repaired Solution Sol	
End	

Table 6.9: Random Repair Process

(2) Greedy Repair

Greedy Repair evaluates the value of each candidate path using the efficiency index 'number of transportation demand / path cost', prioritizing the allocation of vehicles to paths that can serve the most demands at the lowest cost. The algorithm calculates the demand-to-cost ratio for each candidate path to obtain its service value score, selecting the path with the highest input-output efficiency for priority allocation. While the greedy strategy may not guarantee a global optimum, it offers high computational efficiency, enabling the rapid repair of an incomplete solution post-destruction into a complete feasible solution. The specific steps are shown in Table 6.10.

(3) Utility Maximization Repair

Utility Maximization Repair selects paths with higher opportunity costs, where high opportunity cost means that the path achieves a good balance between predicted passenger capacity and cost. For each candidate path in the solution pool that can repair the removed demand, the utility value is first calculated through $Utility_{(p_i)} = \frac{1}{c_{(p_i)}}[(\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}) \times (1 + B_{capacity})]$, where $\gamma_{(p_i)}$ is the maximum number of passengers that can be transported on the path, $\lambda_{(p_i)}$ is the maximum amount of freight that can be transported, $c_{(p_i)}$ is the cost of this path, and $B_{capacity}$ is the capacity utilization rate reward used to expand the advantage of paths with higher average capacity utilization rates. Then the regret value is calculated as $Regret_{(p_i)} = (Utility_{max} - Utility_{(p_i)}) - 0.1 \times \max(0, c_{max} - c_{(p_i)})$, where $Utility_{max}$ is the highest utility value among all paths, and c_{max} is the highest cost among all paths. The regret value is a quantification of decision risk, with lower regret values representing better input-output ratios for this path investment. Finally, a comprehensive score is used to find the optimal trade-off between

Input: Partially Destroyed Solution Sol'

Generate candidate paths which can cover destroyed demand

Iterate while u_e & $u_f \neq 0$:

1 **Calculate** efficiency for each candidate path ($Efficiency(p_i) = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$)

2 **Select** the *highest-efficiency* path

3 **Insert**

Evaluate feasibility

Output: Repaired Solution Sol

End

Table 6.10: Greedy Repair Process

each path's utility and risk control, with the scoring formula being $Utility\ score_{(p_i)} = (0.7 \times \frac{Utility_{(p_i)}}{Utility_{max}}) + (-0.3 \times \frac{Regret_{(p_i)}}{Regret_{max}})$. Paths with higher scores will be preferentially selected and allocated demand in a progressive manner, seeking to allocate the highest-scoring path among the remaining paths that can satisfy unallocated demand. The specific steps are shown in Table 6.11.

Input: Partially Destroyed Solution Sol'

Generate candidate paths which can cover destroyed demand

Iterate while u_e & $u_f \neq 0$:

1 **Calculate** utility for each candidate path:

$$(Utility_{(p_i)} = \frac{1}{c_{(p_i)}}[(\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}) \times (1 + B_{capacity})])$$

2 **Calculate** regret value for each candidate path:

$$(Regret_{(p_i)} = \left(Utility_{max} - Utility_{(p_i)}\right) - 0.1 \times \max(0, c_{max} - c_{(p_i)}))$$

3 **Calculate** utility score for each candidate path:

$$(Utility\ score_{(p_i)} = (0.7 \times \frac{Utility_{(p_i)}}{Utility_{max}}) + (-0.3 \times \frac{Regret_{(p_i)}}{Regret_{max}}))$$

4 **Select** the *highest-scoring* path

5 **Insert**

Evaluate feasibility

Output: Repaired Solution Sol

End

Table 6.11: Utility Maximization Repair

6.5.3. Arc-level repair operator

(1) Arc Insertion Repair

Arc Insertion Repair can find the optimal insertion scheme for removed demand, enabling paths to maintain feasibility while minimizing insertion costs. Each removed arc with demand is assigned a fixed time window, and value ranking is performed through passenger quantity $\gamma_{(a_i)}$ and freight quantity $\lambda_{(a_i)}$. Arcs with more passenger demand and higher value will be preferentially selected, followed by attempting all possible insertion positions in the currently destroyed paths through enumeration. After successfully inserting one position, the remaining demand will continue to attempt insertion into this path. When no suitable arc can be inserted, the lowest-cost non-demand arc is found to repair the path to complete feasibility, calculating the cost required to use this path. After evaluating the insertion

cost of each position, the most efficient scheme is selected, as the path with the highest score in this formula: $Efficiency_{(p_i)} = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$. Since this operator exhaustively searches all possible insertion combinations, it can ensure finding locally optimal solutions. The specific steps are shown in Table 6.12.

Input: Partially Destroyed Solution Sol'

Identify arcs which have destroyed demand & **Assign** each demand arc a time window

Iterate while u_e & $u_f \neq 0$:

1. **Rank** demand arcs by number of passengers $\gamma_{(a_i)}$ and freight $\lambda_{(a_i)}$
2. **Select** *currently highest-ranked* arc:
3. **Enumerate** all possible insertion position by the time window on a partially destroyed path & **Insert**
4. **Continue** searching remaining demand arcs by the time window & **Insert** if the arc fit the time window
5. **Find** lowest-cost non-demand arcs to connect & **Evaluate** feasibility
6. **Collect** all possible repaired paths
7. **Calculate** efficiency for each repaired path ($Efficiency(p_i) = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$)
8. **Select** the *highest-efficiency* path
9. **Insert**

Output: Repaired Solution Sol

End

Table 6.12: Arc Insertion Repair Process

(2) Chain Repair

Chain Repair outputs optimized and reorganized complete paths by inputting all broken arc segments and unsatisfied demand. Its objective is

$\min \sum_{p_i \in P}(\sum_{fragments(p_i)} c_{fragments(p_i)} + \sum_{connections(p_i)} c_{connections(p_i)})$

where $connections(p_i)$ includes arcs with demand as well as non-demand arcs used for connections. This operator selects the path set with the minimum total cost from all feasible combinations, with the capability of combining multiple paths, optimizing from an overall reconstruction perspective. The specific steps are shown in Table 6.13.

Input: Partially Destroyed Solution Sol'

Identify arcs which have destroyed demand & all paths with broken arc segments

1. **Set** the target:

$$min \sum_{p_i \in P'} (\sum_{fragments(p_i)} c_{fragments(p_i)} + \sum_{connections(p_i)} c_{connections(p_i)})$$

2. **Generate** all feasible path sets P'
2. **Select** *minimum total cost* path set P'

Output: Repaired Solution Sol

End

Table 6.13: Chain Repair Process

(3) Hybrid Repair

When an originally short path has a certain number of arcs removed, the length of remaining arc segments may not meet the preservation conditions, leading to the entire path being removed, while some paths are not completely destroyed, resulting in mixed destruction. When mixed destruction occurs, the algorithm can intelligently identify and assign the Hybrid Repair operator for restoration. For paths with existing segments, Arc Insertion Repair is used, and if there is demand that cannot be allocated, path-level Greedy Repair is employed for allocation. This operator can first handle high-certainty demand, then process remaining demand with minimal waste. The specific steps are shown in Table 6.14.

Input: Partially Destroyed Solution Sol'

Identify all paths if broken arc segments exist:

1 Path with segments exist:
  Apply *Arc Insertion Repair*

2 Path with no segment exist:
  Apply *Greedy Repair*

Output: Repaired Solution Sol

End

Table 6.14: Hybrid Repair Process

6.6. Operator Selection Mechanism

The Destroy and Insertion operators adopt a roulette wheel selection strategy, which maintains the weight of each operator to dynamically adjust the selection probability. Specifically, the selection probability of operator i is calculated by the formula $P_i = \frac{w_i}{\sum_{j=1}^{n} w_j}$, where w_i is the current weight of operator i, and $\sum_{j=1}^{n} w_j$ is the sum of all operator weights of the same type. All operators have an initial weight of 1.0, ensuring that each operator has an equal chance of being selected at the beginning of the algorithm. When an operator combination produces an improvement, the corresponding operator weights are updated according to $w_i^{t+1} = w_i^t \times (1 + \alpha)$; when no improvement is produced, the weights are updated according to $w_i^{t+1} = w_i^t \times (1 - 0.5\alpha)$, where α is the weight update factor. This selection mechanism enables superior operator combinations to obtain higher selection probabilities, thereby improving overall search efficiency.

7

Numerical experiments

To verify the performance of the proposed approaches, this section first tests some small-scale data to examine the gap and effectiveness between the algorithms, and then conducts numerical experiments on a real case study. The proposed algorithm is coded in Python on a Windows 11 personal computer with 13th Gen Intel(R) Core(TM) i7-13700H and 32G RAM. GUROBI 12.1.1 is used to solve the MAV Routing Problem model.

7.1. Impact of Space-Time Network scale on different solution methods

The scale of the model is primarily determined by the number of stations and timestamps. An increase in the number of timestamps and stations both leads to an increase in the number of nodes in the Space-Time Network, which in turn causes a rise in the number of paths and, consequently, more demand. This section divides the experiments into two parts: fixed timestamps and fixed station numbers, to compare the solution quality and computational speed of different methods. Additionally, the number of fixed stations will be further divided into two aspects: changes in the number of lines and changes in the number of pickup and delivery stops.

The calculation of costs involves 1 timestamp representing 1 minute. MAU's per-minute travel cost depending on its energy consumption, travel speed, and local electricity price. The average energy consumption of MAU is 22 kWh/100km (Solar Impulse Foundation 2025), and it is assumed to operate at an average speed of 30km/h in urban settings (including driving and stopping). The average price for public charging in the Netherlands is €0.43/kWh (EV Connect 2025). The per-minute electricity cost for MAU can be calculated using the following formula:

$$\text{Cost (€/min)} = \frac{\text{Energy consumption (kWh/100km)} \times \text{Speed (km/h)}}{100} \times \text{Electricity cost (€/kWh)} \quad (7.1)$$

The result is a cost of €0.0473/min. Regarding other parameters, based on the NEXT Company database (NEXT Modular Vehicles 2024), A MAU carriage has a capacity Q of 15 passengers, and this study assumes that 2 pieces of freight occupy the capacity of 1 passenger, as ρ is 2. Therefore, the same carriage can accommodate 30 pieces of freight. Additionally, to prevent MAV formation on the road due to the MAU's composition, the number of coupled units G is limited to 3. Furthermore, due to the limited number of timestamps, the travel time is set to be relatively compact, ranging between 1 and 3 minutes.

To investigate the impact of increasing the number of lines on the performance of different algorithms, the timestamp is fixed at 20. As shown in Table 7.1, the first column represents the parameters adjusted in the experiment, denoted as $T - L - PD$, where T is the number of timestamps, L is the number of fixed bus lines, and this experiment uses 2, 3, and 4 lines, with each line fixed at 8 stations, including 2 depots and 6 intermediate stops. Among them, stops s_2 and s_7 are geographically identical and connected by a transfer arc. The delivery and pick-up stops outside the lines are fixed at 4, denoted

as PD. Figure 7.1 illustrates this space network using 2 lines as an example. The second column in Table 7.1 represents the number of paths to be traversed in the current experiment, the fourth column represents the optimal solution obtained by the algorithm, which indicates the total travel time of all MAUs in this network. The fifth column represents the total operational cost calculated based on the travel time. The sixth and seventh columns represent the time taken to obtain the initial solution and the time spent by the algorithm, respectively. The eighth column represents the total computation time to complete the experiment.

Table 7.1: Comparison of algorithms for different number of lines

Instance (T-L-PD)	Number of paths traversed	Solution method	Objective		Computational time (sec)		
			Travel time (min)	Cost (euro)	Initial solution	ALNS algorithm	Total
20 - 2 - 4	40280	GUROBI	–	–	–	–	>21600.0
		GUROBI+ALNS	154	7.28	204.5	23.9	228.4
		Greedy heuristic+ALNS	162	7.66	0.3	28.3	28.6
20 - 3 - 4	67327	GUROBI	–	–	–	–	–
		GUROBI+ALNS	198	9.37	471.3	33.2	504.5
		Greedy heuristic+ALNS	214	10.12	0.4	52.6	53.0
20 - 4 - 4	84301	GUROBI	–	–	–	–	–
		GUROBI+ALNS	290	13.72	1181.7	184.4	1366.1
		Greedy heuristic+ALNS	328	15.51	0.9	201.5	202.4

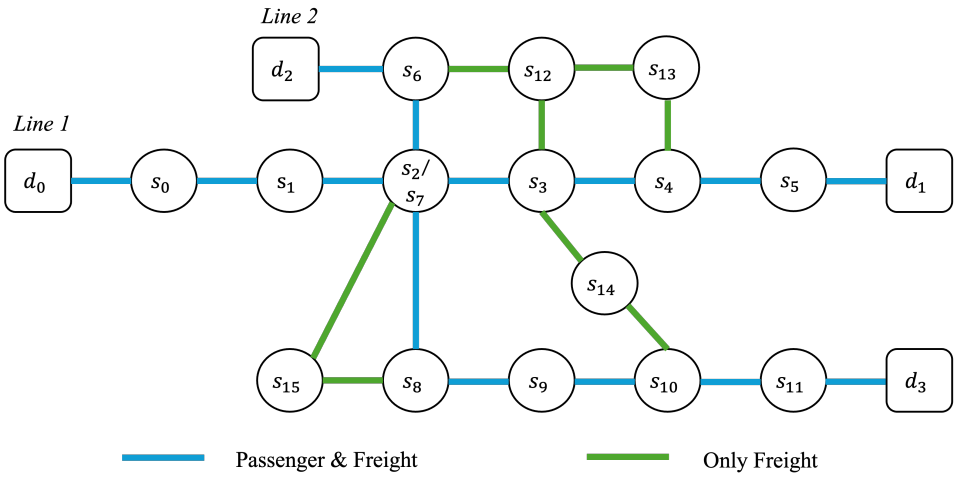

Figure 7.1: Spatial network for experiments with fixed timestamp numbers (20-2-4)

Due to the large number of stations and the dispersed nature of demand, the GUROBI solver struggles to find any solution within 6 hours for a network with 2 lines, 15 passenger demands, and 5 freight demands. In terms of initial solution computation time, the Greedy heuristic method is significantly faster than the GUROBI method, which solves a small-scale path problem with demand. However, the initial solution quality of the GUROBI method is better, leading to shorter subsequent computation times for ALNS. The computation time of the GUROBI method increases exponentially with the number and dispersion of demands, resulting in a growing gap in total computation time compared to the Greedy heuristic + ALNS algorithm as the network size expands. In addition, in the 20-2-4 experiment, the gap between the optimal solutions obtained by the two methods was 5.2%, while in the 20-4-4 experiment, the gap reached 10.3%. The solution quality obtained by the GUROBI+ALNS algorithm is consistently better than that of the Greedy heuristic+ALNS algorithm.

Next, the experiment fixes the number of lines at 2, still using 20 timestamps, with the number of pick-up and delivery stops set to 2, 4, and 6 cases, while the passenger demand remains unchanged, and the

freight demand increases proportionally with the number of pick-up and delivery stops. For example, the connection scenario with 4 pick-up and delivery stops uses the same spatial network as the 20-2-4 configuration, as shown in Figure 7.1. As shown in Table 7.2, since the stops outside these fixed lines can serve to connect different lines, compared to the stops on the fixed lines, each additional pick-up and delivery stop causes the number of paths to be traversed to grow exponentially.

Table 7.2: Comparison of algorithms for different number of pick-up and delivery stops

Instance (T-L-PD)	Number of paths traversed	Solution method	Objective		Computational time (sec)		
			Travel time (min)	Cost (euro)	Initial solution	ALNS algorithm	Total
20 - 2 - 2	6876	GUROBI	134(78.4)	6.34	–	–	1714.6
		GUROBI+ALNS	136	6.43	19.2	8.5	31.7
		Greedy heuristic+ALNS	140	6.62	0.2	9.1	9.3
20 - 2 - 4	40280	GUROBI	–	–	–	–	>21600.0
		GUROBI+ALNS	154	7.28	204.5	23.9	228.4
		Greedy heuristic+ALNS	162	7.66	0.3	28.3	28.6
20 - 2 - 6	92478	GUROBI	–	–	–	–	–
		GUROBI+ALNS	208	9.84	2099.1	87.1	2186.2
		Greedy heuristic+ALNS	237	11.2	2.3	113.1	115.4

In the smallest-scale experiment, the GUROBI solver obtained the optimal solution, with the GUROBI+ALNS method achieving an optimal solution with a gap of 1.5%, and the Greedy heuristic+ALNS method at 4.5%. Combined with the previous experiment that varied the number of lines, it can be observed that GUROBI is highly sensitive to the spatial network's expansion in terms of computational speed. When using GUROBI to generate initial solutions, it is evident that adding two more pick-up and delivery stops increases the computation time by approximately tenfold. In contrast, the heuristic methods are less affected. To further validate the gap with the exact solutions obtained by the GUROBI solver, the number of stations was reduced to decrease the dispersion of demand across the spatial network.

To further verify the gap between the exact solution obtained by the GUROBI solver, the number of timestamps was reduced to decrease the spatial dispersion of demand. A fixed number of stations is used, and the algorithm's quality was tested by progressively increasing the number of timestamps. As shown in Figure 7.2, the experiment utilized two fixed lines. Additionally, there were two delivery and pick-up stops, namely s_{12} and s_{13}, where s_{12} can connect to s_3 and s_6, and s_{13} can connect to s_2, s_7, and s_8. As shown in Table 7.3, this experiment used 12 groups of tests, with timestamps ranging from 10 to 120, increasing by intervals of 10.

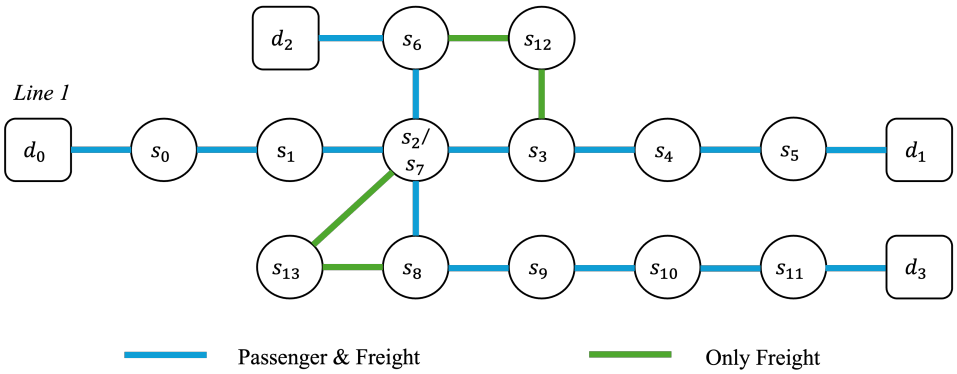

Figure 7.2: Spatial network for fixed station number experiments

Taking the experiment with 10-2-2 as an example, Table 7.4 shows the paths assigned to each MAU in the Space-Time network and the demand on each corresponding arc. From the results, it can be observed that some demands are split, or the remaining capacity of the carriages is utilized for freight

Table 7.3: Comparison of algorithms for different number of timestamps

Instance (T-L-PD)	Number of paths traversed	Solution method	Objective		Computational time (sec)		
			Travel time (min)	Cost (euro)	Initial solution	ALNS algorithm	Total
10 - 2 - 2	153	GUROBI	69(56.7)	3.26	–	–	2.2
		GUROBI+ALNS	69	3.26	1.1	2.8	3.9
		Greedy heuristic+ALNS	71	3.45	0.2	3.1	3.2
20 - 2 - 2	26905	GUROBI	102(69.0)	4.82	–	–	1440
		GUROBI+ALNS	103	4.92	23.6	20.5	44.1
		Greedy heuristic+ALNS	107	5.20	0.3	27.3	27.6
30 - 2 - 2	250244	GUROBI	–	–	–	–	$>$21600.0
		GUROBI+ALNS	155	7.33	48.7	216.6	265.3
		Greedy heuristic+ALNS	155	7.33	0.4	233.6	234.0
40 - 2 - 2	429590	GUROBI	–	–	–	–	–
		GUROBI+ALNS	159	7.52	82.2	297.5	379.5
		Greedy heuristic+ALNS	163	7.71	0.6	420.1	420.7
50 - 2 - 2	654375	GUROBI	–	–	–	–	–
		GUROBI+ALNS	180	8.51	538.6	394.6	933.2
		Greedy heuristic+ALNS	187	8.85	1.1	602.8	603.9
60 - 2 - 2	840069	GUROBI	–	–	–	–	–
		GUROBI+ALNS	212	10.03	630.2	662.6	1292.8
		Greedy heuristic+ALNS	215	10.17	1.3	788.1	789.4
70 - 2 - 2	1049975	GUROBI	–	–	–	–	–
		GUROBI+ALNS	244	11.54	1085.0	691.5	1776.5
		Greedy heuristic+ALNS	258	12.20	1.7	1403.9	1405.6
80 - 2 - 2	1247367	GUROBI	–	–	–	–	–
		GUROBI+ALNS	292	13.81	1864.5	704.8	2569.3
		Greedy heuristic+ALNS	303	14.33	2.3	2040.6	2042.9
90 - 2 - 2	1463988	GUROBI	–	–	–	–	–
		GUROBI+ALNS	376	17.78	2068.9	1187.6	3256.5
		Greedy heuristic+ALNS	391	18.49	3.1	2543.8	2546.9
100 - 2 - 2	1677193	GUROBI	–	–	–	–	–
		GUROBI+ALNS	402	19.01	3249.7	1781.5	5031.2
		Greedy heuristic+ALNS	435	20.58	4.0	3367.8	3371.8
110 - 2 - 2	1859714	GUROBI	–	–	–	–	–
		GUROBI+ALNS	424	20.06	4953.2	2414.9	7368.1
		Greedy heuristic+ALNS	458	21.66	4.8	3750.7	3755.5
120 - 2 - 2	2073843	GUROBI	–	–	–	–	–
		GUROBI+ALNS	445	21.05	10772.5	2987.1	13759.6
		Greedy heuristic+ALNS	460	21.79	7.6	6876.9	6884.5

transportation. Regarding the composition of MAVs, for instance, k_1 and k_2 form an MAV in some segments of the path, while k_3 and k_4, for example, travel together throughout the entire journey. Additionally, for example, k_9 does not perform any transportation tasks and is an MAU traveling empty to balance depot flow.

In small-scale tests, the GUROBI solver is able to provide exact solutions. In the Objective column, the exact integer solution obtained by the GUROBI solver is accompanied by the relaxed solution in parentheses. In two sets of experiments, the gaps between these two solutions were 17.9% and 32.4%, respectively, fully demonstrating the complexity of the model. Through comparison, the solution quality of both the GUROBI+ALNS algorithm and the Greedy heuristic method+ALNS algorithm fell within an acceptable range, as shown in Figure 7.3. In the 10-2-2 experiment, the gap between the GUROBI+ALNS algorithm's solution and GUROBI's solution was 0.0%, while the gap for the Greedy

Table 7.4: Experimental results example (10-2-2)

MAU id and Demand	Arc: start station (start timestamp) → end station (end timestamp)						
k_0	$d_0(0) \rightarrow s_0(2)$	$s_0(2) \rightarrow s_1(4)$	$s_1(4) \rightarrow s_2(6)$	$s_2(6) \rightarrow d_1(8)$			
Passenger demand	$e_0 : 10$	-	-	$e_3 : 15$	-	-	-
Freight demand	$f_0 : 10$	$f_1 : 30$	-	-	-	-	-
k_1	$d_0(0) \rightarrow s_0(2)$	$s_0(2) \rightarrow s_1(4)$	$s_1(4) \rightarrow s_2(6)$	$s_2(6) \rightarrow d_1(8)$			
Passenger demand	$e_0 : 10$	$e_1 : 15$	$e_2 : 15$	$e_3 : 15$	-	-	-
Freight demand	-	-	-	-	-	-	-
k_2	$d_2(1) \rightarrow s_3(2)$	$s_3(2) \rightarrow s_4(4)$	$s_4(4) \rightarrow s_5(6)$	$s_5(6) \rightarrow d_3(8)$			
Passenger demand	$e_7 : 10$	$e_8 : 5$	$e_{10} : 5$	$e_1 1 : 5$	-	-	-
Freight demand	-	$f_2 : 20$	$f_5 : 10$	$f_6 : 10$	-	-	-
k_3	$d_0(2) \rightarrow s_0(3)$	$s_0(3) \rightarrow s_6(4)$	$s_6(4) \rightarrow s_0(5)$	$s_0(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_4(7)$	$s_4(7) \rightarrow s_3(8)$	$s_3(8) \rightarrow d_2(9)$
Passenger demand	$e_{12} : 15$	-	-	$e_9 : 10$	$e_{14} : 10$	-	-
Freight demand	-	$f_7 : 10$	$f_8 : 10$	$f_9 : 10$	-	-	$f_{10} : 20$
k_4	$d_0(2) \rightarrow s_0(3)$	$s_0(3) \rightarrow s_6(4)$	$s_6(4) \rightarrow s_0(5)$	$s_0(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_4(7)$	$s_4(7) \rightarrow s_3(8)$	$s_3(8) \rightarrow d_2(9)$
Passenger demand	$e_{12} : 10$	$e_{13} : 10$	-	$e_9 : 15$	$e_{14} : 5$	$e_{15} : 5$	$e_{16} : 5$
Freight demand	-	-	-	-	-	-	$f_{10} : 10$
k_5	$d_2(3) \rightarrow s_3(4)$	$s_3(4) \rightarrow s_4(5)$	$s_4(5) \rightarrow s_5(6)$	$s_5(6) \rightarrow d_3(8)$			
Passenger demand	-	-	-	$e_8 : 15$	-	-	-
Freight demand	-	-	$f_4 : 25$	-	-	-	-
k_6	$d_1(3) \rightarrow s_2(5)$	$s_2(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$			
Passenger demand	$e_4 : 5$	-	$e_5 : 5$	-	-	-	-
Freight demand	-	$f_{13} : 10$	-	-	-	-	-
k_7	$d_3(3) \rightarrow s_5(5)$	$s_5(5) \rightarrow s_4(6)$	$s_4(6) \rightarrow s_1(7)$	$s_1(7) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$		
Passenger demand	-	-	-	-	-	-	-
Freight demand	-	$f_{12} : 15$	$f_{11} : 30$	$f_{14} : 10$	$f_3 : 5$	-	-
k_8	$d_3(3) \rightarrow s_5(5)$	$s_5(5) \rightarrow s_4(6)$	$s_4(6) \rightarrow s_1(7)$	$s_1(7) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$		
Passenger demand	$e_{19} : 5$	$e_{17} : 5$	-	$e_{18} : 10$	$e_{20} : 5$	-	-
Freight demand	-	$f_{12} : 20$	$f_{11} : 10$	-	-	-	-
k_9	$d_1(3) \rightarrow s_2(5)$	$s_2(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$			
Passenger demand	$e_4 : 15$	-	-	$e_6 : 10$	-	-	-
Freight demand	-	-	-	$f_3 : 10$	-	-	-

heuristic method+ALNS algorithm was 2.9%. In the 20-2-2 experiment, the gap for the GUROBI+ALNS algorithm was 0.9%, while the gap for the Greedy heuristic method+ALNS algorithm was 4.7%. Although the GUROBI+ALNS algorithm requires more time to solve, its solution quality is superior, closely approaching the exact solution. However, starting from the 30-2-2 experiment, due to the excessive number of paths and demands to be searched, the GUROBI solver failed to produce any results within 6 hours. In contrast, the Greedy heuristic method+ALNS algorithm was able to find a result in 234 seconds. While this method significantly improves computational speed, it sacrifices some solution quality.

As the number of timestamps and demands increases, as shown in Figure 7.4, the computational time gap between the GUROBI+ALNS algorithm and the Greedy heuristic method+ALNS algorithm be-

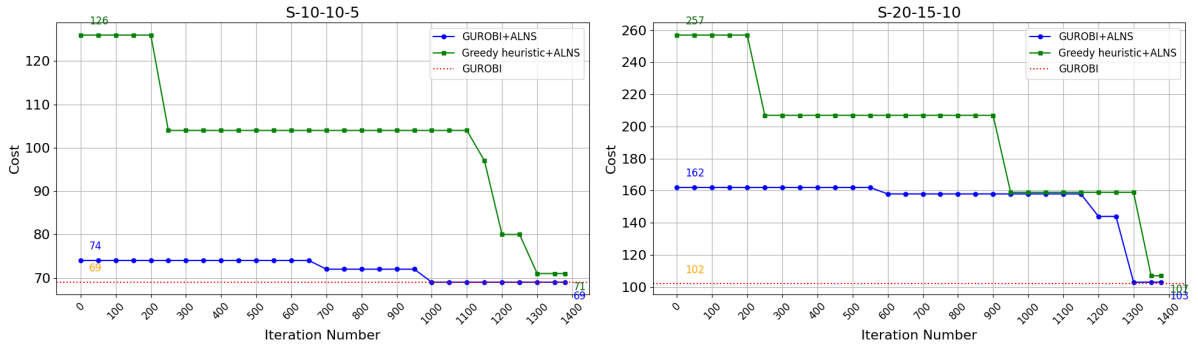

Figure 7.3: Comparison of the quality of the optimal solution of the algorithms

comes increasingly significant. In the 70-2-2 experiment, the total time difference between the two methods was 370.9 seconds, while in the 100-2-2 experiment, the time difference expanded to 1659.4 seconds. In the largest-scale experiment, 120-2-2, the time difference between the two methods reached 6875.1 seconds, with the computational time of the GUROBI+ALNS algorithm nearly double that of the Greedy heuristic method+ALNS algorithm. It can be predicted that when the number of stations is fixed, an increase in the number of timestamps will lead to a progressively larger computational time gap between the two methods.

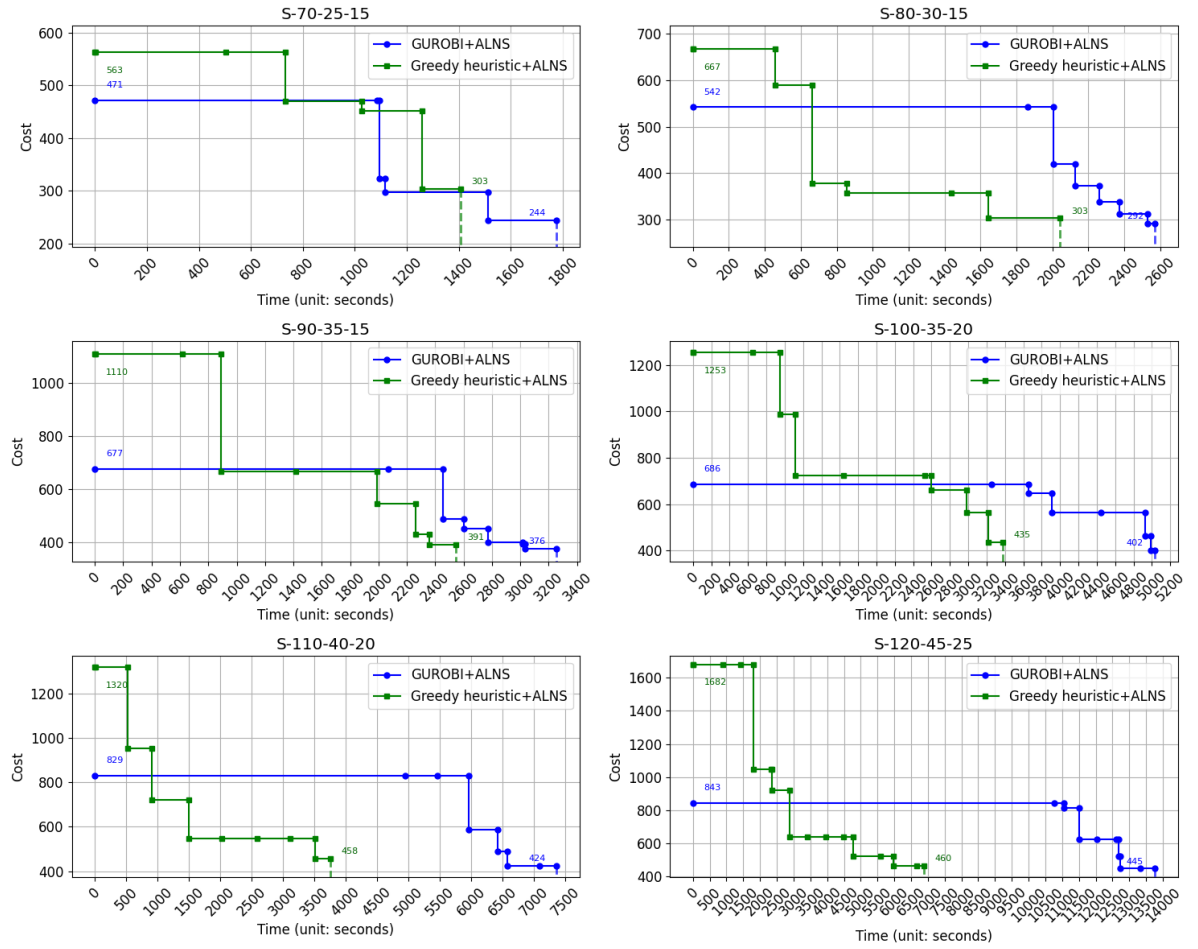

Figure 7.4: Comparison of algorithms computational time

By combining the results of experiments with a fixed number of timestamps and stations, the following observations can be made:

1. Compared with the exact optimal solutions obtained by the GUROBI solver in small-scale experiments, the solutions from the GUROBI+ALNS algorithm are very close to the exact solutions, while the solutions from the Greedy heuristic method+ALNS algorithm exhibit a slightly larger gap but remain within an acceptable range.
2. When the number of timestamps is fixed, increasing the number of stops in fixed lines, pick-up and delivery stops, and demands leads to a progressively more noticeable superiority in the solution quality of the GUROBI+ALNS algorithm compared to the Greedy heuristic method+ALNS algorithm. However, when generating initial solutions, the computational time of the GUROBI method increases exponentially with the number of stations, especially when the number of pick-up and delivery stops increases. In contrast, the computational speed advantage of the Greedy heuristic method+ALNS algorithm will become increasingly evident.
3. When the number of stations is fixed, increasing the number of timestamps and demands does not lead to a significant gap in solution quality between the GUROBI+ALNS algorithm and the Greedy heuristic method+ALNS algorithm. The overall computational time for both methods increases steadily, but the Greedy heuristic method+ALNS algorithm remains faster, with the gap widening as the number of timestamps increases.
4. In the Space-Time Network of the MAU Routing Problem, both methods' computational speed of generating initial solutions is influenced by the number of paths to traverse and the number of demands. When expanding through the spatial dimension, the computational time growth rate for both the Greedy heuristic initialization method and the GUROBI initialization method is significantly faster than when expanding through the temporal dimension. This is because, under the influence of the time window of freight demands, a greater number of stations, especially pick-up and delivery stops, leads to a faster increase in path diversity compared to a greater number of timestamps, resulting in a more significant increase in demand paths. The GUROBI initialization method requires more time, which limits the scale of experiments using this method to some extent. However, since the Greedy heuristic initialization method quickly covers demands based on path scores, the increase in computational time is much less noticeable.
5. The computational time of the ALNS algorithm is insensitive to either dimension (spatial or temporal) and is mostly affected by the quality of the initial solution. Higher-quality initial solutions accelerate the algorithm's convergence process and avoid repeated repair processes due to failed feasibility checks, resulting in shorter computational times. Consequently, the ALNS algorithm using the GUROBI initialization method typically converges faster than when using the Greedy heuristic initialization method. In summary, the ALNS algorithm tailored for the MAU Routing Problem can converge within a limited number of iterations in all instances.

7.2. Sensitivity analysis of passenger and freight capacity occupancy ratio

The passenger and freight capacity occupancy ratio ρ represents the size of the freight, where a larger ρ indicates that the freight occupies less space within the carriage. This section uses a 60-2-2 network to perform a sensitivity analysis on ρ. The cost variations are examined for ρ values of 2, 3, 4, 5, and 6, under the condition that passenger demand and freight demand remain constant and Q is set to 15. As shown in Table 7.5, the required number of MAUs and the cost results obtained from two initialization methods are recorded.

Figure 7.5 illustrates the trend of cost changes under different passenger and freight capacity occupancy ratios. As ρ increases, more freight can be accommodated within the MAU space. Freight that previously required multiple MAUs for transport can now be handled by fewer MAUs, leading to a gradual reduction in the number of MAUs used and, consequently, a decrease in the additional travel costs associated with MAU usage. When $\rho = 4$, a significant reduction is observed, with the number of MAUs used decreasing by 4 compared to when $\rho = 2$, resulting in a cost reduction of 16.1%. This is primarily because, in this experiment, freight demand frequently consists of around 45 to 50 units, while demands exceeding 50 units are rare. Consequently, when $\rho = 5$ or $\rho = 6$, the optimization results show no further cost reduction. The changes in MAU quantity and operational costs caused by ρ primarily depend on the freight demand between pairs of stations and secondarily on the passenger

Table 7.5: Result of sensitivity analysis of passenger and freight capacity occupancy ratio

ρ	Solution method	Number of MAU used	Travel time (min)	Cost (euro)
2	GUROBI+ALNS	19	212	10.03
	Greedy heuristic+ALNS	19	215	10.17
3	GUROBI+ALNS	18	203	9.60
	Greedy heuristic+ALNS	19	207	9.79
4	GUROBI+ALNS	15	178	8.42
	Greedy heuristic+ALNS	15	182	8.61
5	GUROBI+ALNS	14	174	8.23
	Greedy heuristic+ALNS	15	181	8.56
6	GUROBI+ALNS	14	174	8.23
	Greedy heuristic+ALNS	15	180	8.51

demand within the time window. If the passenger demand on the arcs between two stations within the freight transport time window is consistently high, additional MAUs are still required to transport freight even when ρ is large, making it challenging to achieve significant reductions in optimized operational costs.

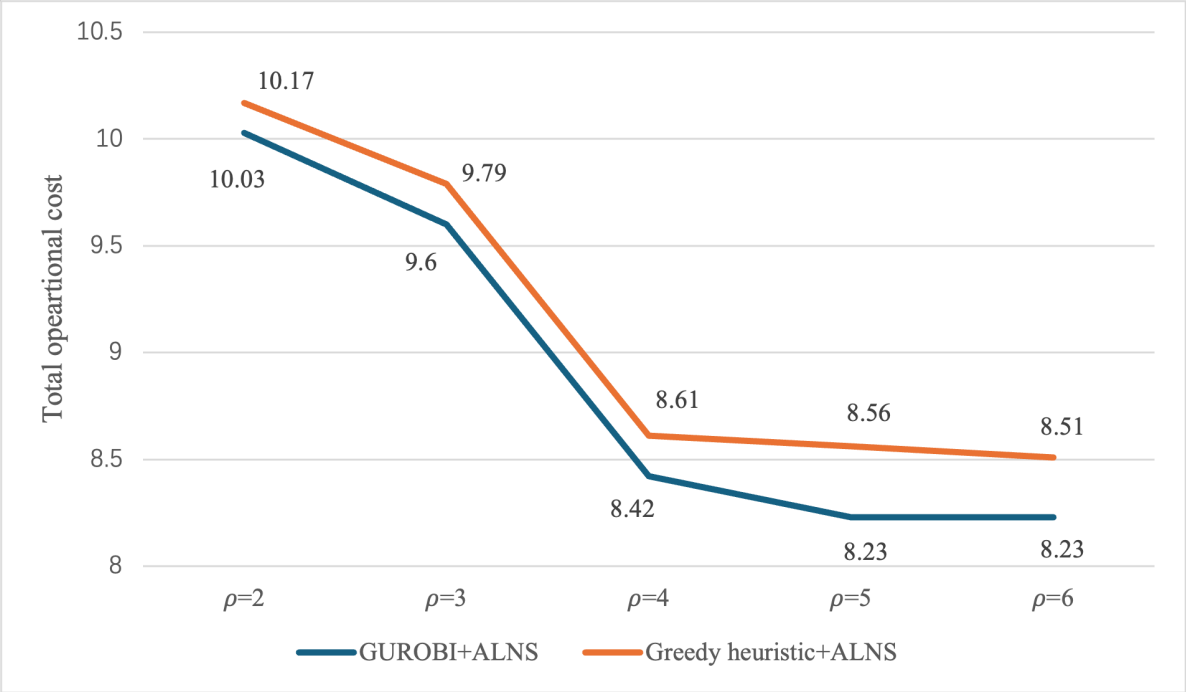

Figure 7.5: Operating costs at different passenger and freight capacity occupancy ratios

7.3. Sensitivity analysis of operators

Among all customized operators, the path-level Utility Maximization Repair operator and the arc-level Chain Repair operator are logically complex, involving multiple layers of functions. The Utility Maximization Repair method requires multiple traversals to evaluate candidate paths, iteratively calculating path utility, capacity utilization, and regret values. Conversely, the Chain Repair operator attempts to combine all remaining path segments into complete paths, involving repetitive path connection checks and demand allocation optimization. These two operators play unique roles in exploring complex solution spaces and optimizing path combinations, but their intricate processes can increase the computational time of the entire ALNS algorithm. To assess the contribution of these two insertion operators to the

overall solution quality and to balance with computational speed, the following experiments were designed.

The experiments were conducted using a spatial layout same as the fixed number of stations, and three groups of experiments with timestamp counts of 50, 60, and 70 were set up for mutual comparison, focusing on the computational time and solution quality changes of the ALNS algorithm. The nine operators other than the Utility Maximization Repair operator and the Chain Repair operator were collectively referred to as basic operators. Table 7.6 and Table 7.7 respectively present the computational time and optimal solutions for the GUROBI + ALNS method and the Greedy heuristic + ALNS method under different operator combination conditions.

Table 7.6: Sensitivity analysis of operators for GUROBI + ALNS method

Operator selection	50-2-2		60-2-2		70-2-2	
	Calculation time (sec)	Objective (Travel time)	Calculation time (sec)	Objective (Travel time)	Calculation time (sec)	Objective (Travel time)
ALNS with basic operators	217.8	225	434.4	265	481.2	305
ALNS with basic operators + Utility Maximization Repair operator	271.9	198	495.6	248	501.9	287
ALNS with basic operators + Chain Repair operator	293.2	182	531.3	220	539.6	249
ALNS with full operators	394.6	180	662.6	212	691.5	244

Table 7.7: Sensitivity analysis of operators for Greedy heuristic + ALNS method

Operator selection	50-2-2		60-2-2		70-2-2	
	Calculation time time (sec)	Objective (Travel time)	Calculation time (sec)	Objective (Travel time)	Calculation time (sec)	Objective (Travel time)
ALNS with basic operators	332.5	243	434.4	279	772.1	335
ALNS with basic operators + Utility Maximization Repair operator	415.4	222	543.6	237	965.3	285
ALNS with basic operators + Chain Repair operator	448.8	188	616.3	213	1142.0	261
ALNS with full operators	602.8	187	788.1	215	1403.9	258

In terms of computational time, removing these two operators allows the algorithm to complete in nearly half the original time. In the same cases, the Chain Repair operator, which is more complex for global optimization compared to the Utility Maximization Repair operator, often requires more computational time, but the solution quality is very close to that obtained using all operators. In the 60-2-2 experiment with the Greedy heuristic + ALNS method, the solution quality even surpassed that of using all operators, with the largest gap across all experiments being 3.6%. In contrast, although the Utility Maximization Repair operator requires less computational time, its solution quality is poorer, with an average gap of around 9.5% compared to the solution using full operators. Additionally, in several experiments,

it exhibited lower weight usage, but still provided some improvement over solution using only basic operators.

The experimental results indicate that omitting the Chain Repair operator significantly reduces solution quality. Although using the complete set of operators generally yields better near-optimal solutions in nearly all cases, removing these two operators can be considered when decisions need to be made in a shorter time. In large-scale problems, retaining the Chain Repair operator is more beneficial for maintaining solution quality. As the problem scale increases, the combination of basic operators and the Chain Repair operator saves more time per iteration, making it worthwhile to trade a small portion of solution quality for a 20% reduction in computational time.

7.4. Case study

This section validates the advantages of MAVs compared to traditional buses and delivery vans in the real road network and actual demand scenario of Changning District, Shanghai, China. The spatial network of this study is illustrated in Figure 7.6. Two regional bus lines were selected, where Line 72 consists of 8 stations $(d_0, s_0, s_1, s_2, s_3, s_4, s_5, d_1)$, and Line 54 consists of 7 stations $(d_2, s_6, s_7, s_8, s_9, s_{10}, d_3)$. Additionally, there are 4 pick-up and delivery stops outside the fixed lines $(s_{11}, s_{12}, s_{13}, s_{14})$. s_2 and s_7 are the intersection points of the two lines, connected via a transfer arc. Among the pick-up and delivery stops outside the fixed lines, s_{11} and s_{12} are interconnected, with s_{11} connected to s_9 and s_{12} connected to s_3. Similarly, s_{13} and s_{14} are interconnected, with s_{13} connected to s_0 and s_1, and s_{14} connected to s_8.

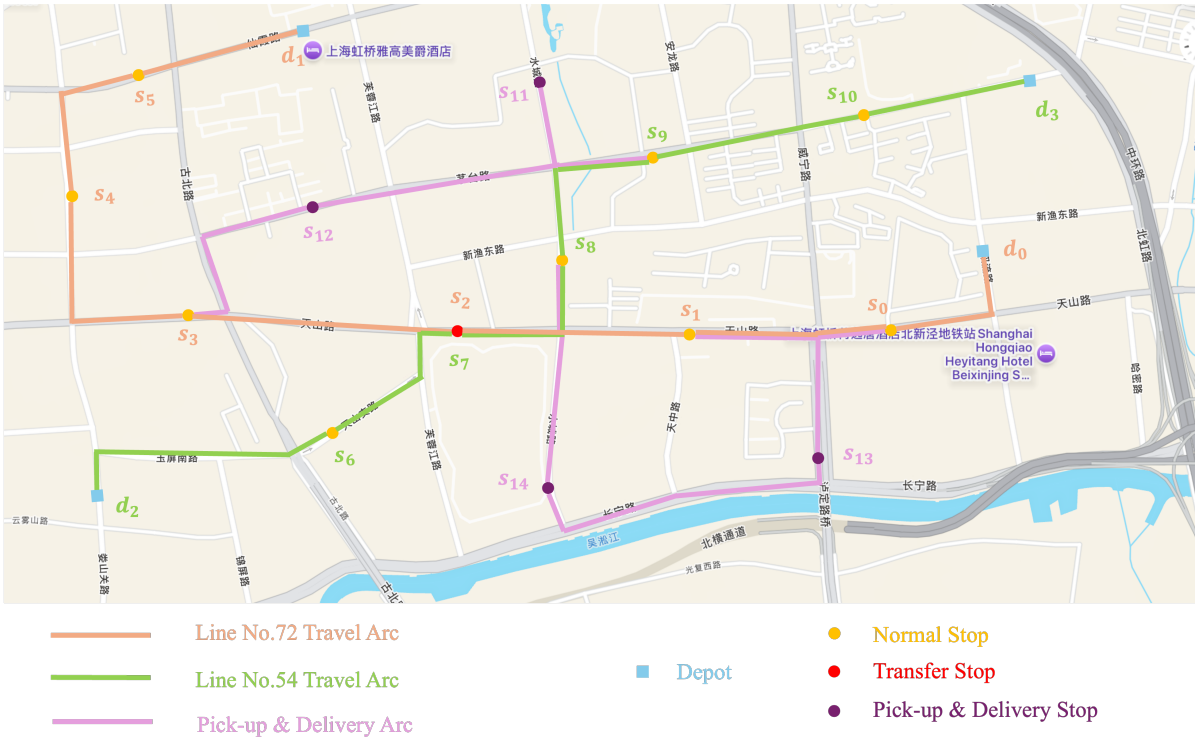

Figure 7.6: Spatial network of the case study MAV route in Shanghai region

The study selects the morning period from 8:00 to 9:30 as the temporal network, comprising 90 timestamps. The peak period is from 8:00 to 9:00, while the off-peak period is from 9:00 to 9:30. For the same distance, the travel time during the peak period ranges from 4 to 7 minutes, whereas during the off-peak period, it reduces to 2 to 5 minutes. For the fixed lines, the bus timetable operates at a frequency of every 15 minutes during the peak period (8:00, 8:15, 8:30, 8:45) and every 20 minutes during the off-peak period (9:00). Idle MAVs can depart at any time to assist in fulfilling the remaining passenger demand and freight demand. Passenger demand data were obtained from statistics provided by the Shanghai Municipal Transportation Commission. Freight demand data were estimated based on

the regional parcel throughput provided by Cainiao with the time window for freight demand spanning all 90 timestamps.

This case study continues to use the parameter information from Section 1, where $Q = 15$, $\rho = 2$, and $G = 3$. As shown in Table 7.8, due to the wide range of the demand distribution, the GUROBI solver was unable to find any solution within 12 hours. The GUROBI initialization method, even with a reduced number of paths, still failed to produce any results within 8 hours. Therefore, subsequent analyses are based on the solutions obtained from the Greedy heuristic + ALNS algorithm.

Table 7.8: Case Result

Solution method	Objective travel time (min)	Initial solution time (sec)	Algorithm time (sec)	Total computational time (sec)
GUROBI	–	–	–	$>$43200
GUROBI+ALNS	–	$>$28800	–	–
Greedy heuristic+ALNS	1054	28.1	8058.2	8086.3

A total of 30 MAUs are deployed in this operation. The detailed path assignments for each MAU are presented in Appendix A. As illustrated in Figure 7.7, freight transportation is seldom selected during peak hours due to the overwhelming passenger demand. During off-peak periods, each MAU has more remaining capacity, which can be better utilized by freight. During off-peak periods, traditional bus system, in order to balance passenger waiting times, find it difficult to significantly reduce the frequency of departures, resulting in a large amount of redundant capacity in the vehicles. To better demonstrate the flexible utilization of space by MAU, Figure 7.8, taking k_{25} as an example, illustrates the entire process of fulfilling both partial passenger demand and freight demand.

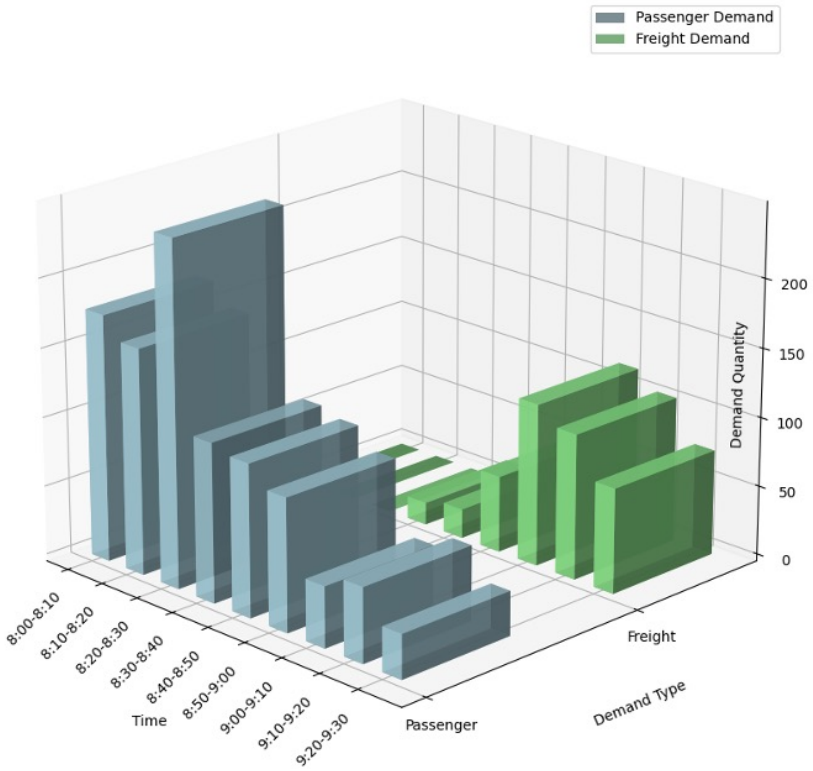

Figure 7.7: Distribution of demand over time

If a co-transportation transit system is not used and a strategy of transporting passengers and freight separately is adopted instead, 38 MAUs are required, with the specific path assigned to each MAU

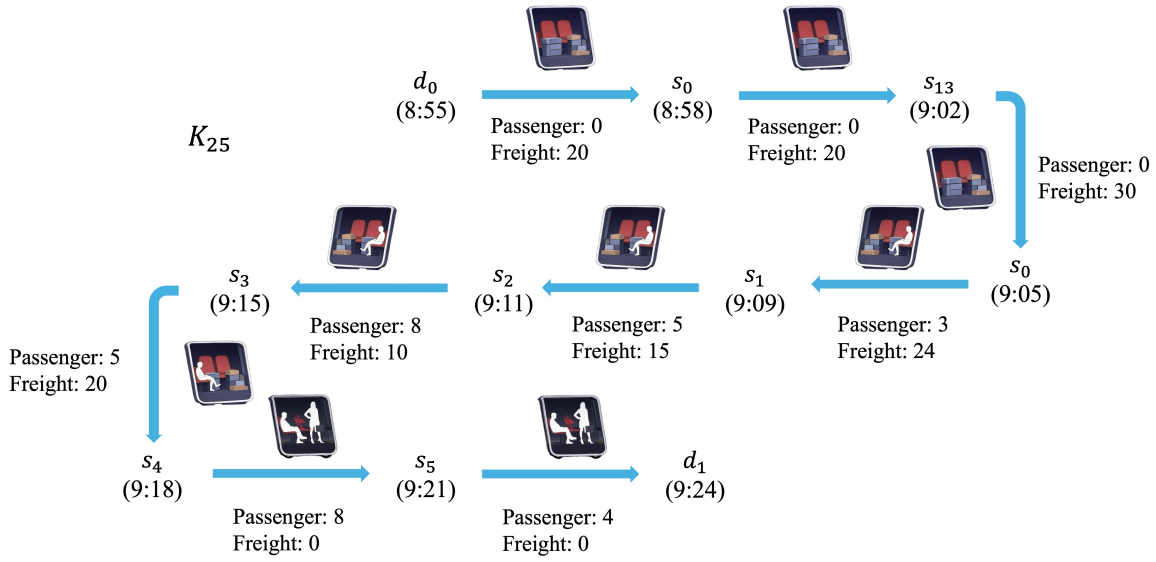

Figure 7.8: Example of MAU flexible transport

detailed in Appendix B. Since the passenger demand on an arc during off-peak periods typically does not exceed 10 people, a significant amount of idle capacity arises. This capacity is not fully utilized by freight, leading to an increase in the number of vehicles used and a corresponding rise in operational costs.

The electricity cost for one MAU in China is €0.036 per minute. For comparison, it is assumed that traditional buses and delivery vans are also electrically powered, with the same electricity cost of €0.089 per minute. According to the bus timetable (Bendibao 2025), a total of 5 departures are required from each of the 4 depots, with 4 departures during peak periods, each taking an average of 35 minutes to complete a transport task, and 1 departure during off-peak periods, each taking an average of 28 minutes to complete a transport task. The total time required to complete all tasks is 672 minutes. For freight demand, delivery vans are used to meet all freight requirements, departing from 4 depots to serve a total of 15 stations with freight demand. The solution is obtained using the Traveling Salesman Problem (TSP) model, with the objective of minimizing the travel time through all demand points, where the travel time is based on the average transit time of each arc.

$$\min \quad \sum_{i=0}^{14} \sum_{j=0, j\neq i}^{14} t_{ij} x_{ij} \tag{7.2a}$$

$$\text{s.t.} \quad \sum_{j=0, j\neq i}^{14} x_{ij} = 1 \quad \forall i \in \{0, 1, \ldots, 14\} \tag{7.2b}$$

$$\sum_{i=0, i\neq j}^{14} x_{ij} = 1 \quad \forall j \in \{0, 1, \ldots, 14\} \tag{7.2c}$$

$$u_i - u_j + 15 x_{ij} \leq 14 \quad \forall i, j \in \{1, 2, \ldots, 14\}, i \neq j \tag{7.2d}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in \{0, 1, \ldots, 14\}, i \neq j \tag{7.2e}$$

$$u_i \in [1, 14] \quad \forall i \in \{1, 2, \ldots, 14\} \tag{7.2f}$$

According to the solution obtained from the GUROBI solver, one delivery van departs from each of the four depots, with a total minimum time of 142 minutes to complete all freight demand. Since both traditional buses and delivery vans require drivers, the cost must include the salaries of all drivers (Shanghai Municipal Human Resources and Social Security Bureau 2025), with an average hourly

wage of €6.5 per driver. As bus drivers can continue to perform return transport tasks after reaching the depot, a total of 12 bus drivers and 4 delivery van drivers are required.

Table 7.9: Cost Comparison between MAU and traditional bus & delivery van

Transport Method	Objective travel time (min)	Electricity cost (euro)	Labor cost (euro)	Total Operating cost (euro)
MAU co-transportation	1054	37.94	–	37.94
MAU separate transportation	1097	39.49	–	39.49
Traditional bus + delivery van	814	72.45	120.37	228.45

As shown in Table 7.9, the MAU co-transportation system offers significant operational cost advantages. Within this one-and-a-half-hour period case, it saves €1.52 compared to the separate transportation system and €190.48 compared to a traditional bus and delivery van system. Compared to a modular transit system without passenger and freight co-transportation, the cost difference in this case is only 3.9%. However, as freight demand and scale of instances increase, this could lead to considerable cost inefficiencies. In contrast, the cost difference with the traditional transportation system reaches 83.4%. Using MAUs not only reduces labor costs but also leverages the smaller capacity of each MAU compartment. During peak periods, MAUs transport passengers, and during off-peak periods, when passenger demand on an arc is low, they can utilize idle capacity to transport freight, significantly reducing capacity waste. Although the total travel time of the traditional transportation system is shorter, the larger vehicle size results in higher unit electricity consumption, leading to electricity costs nearly double those of the MAU system.

8

Conclusion

This study proposes a passenger and freight co-transportation system utilizing Modular Autonomous Units (MAUs) to fulfill all transportation demands on a network with minimal electricity consumption operational costs. MAUs can couple/decouple on any route segment, providing passenger and freight transportation services on multiple fixed bus lines (FRT), as well as freight transportation services between pick-up and delivery stops generated by daily express demands that are not on fixed lines (DRT). Through transfer arcs and delivery arcs, each MAU can move between different lines. A network is established using Space and Time dimensions, where each passenger demand has precise station and time information, and each freight demand is assigned a set of time windows. Based on this, a path-based model for the MAU Routing Problem is developed to satisfy every demand task in the network. A customized Adaptive Large Neighborhood Search (ALNS) algorithm, dividing the destruction and repair processes into path-level and arc-level, is employed to tackle the computational challenges of this complex network problem.

Due to the expansion of the overall network in both spatial and temporal dimensions, the demand quantity also increases accordingly. This study provides two methods for generating initial solutions—GUROBI and Greedy heuristic—which, when combined with the ALNS algorithm, can significantly reduce the computation time required to obtain the optimal solution. In small-scale experiments, compared to the exact optimal solution obtained by the GUROBI solver, the solution quality of both methods is proven to be within an acceptable range. The GUROBI-based algorithm deviates by no more than 1.5%, while the Greedy heuristic algorithm, despite greatly improving computation speed, deviates by no more than 5%. Computational results based on the Shanghai regional network demonstrate the advantages of the ALNS algorithm. Compared to traditional buses and delivery vans, MAUs can efficiently utilize vehicle space to transport both passengers and freight simultaneously, significantly reducing the additional operational costs caused by idle capacity within compartments. Without using a co-transportation system, even when employing MAUs for transportation, an additional 3.9% operational cost is incurred. In contrast, compared to traditional transportation systems, operational costs are reduced by 83.4%. Therefore, the use of this co-transportation modular transit system provides operators with an opportunity to reduce operational costs while meeting all transportation demands.

Overall, these findings confirm that the use of a passenger and freight co-transportation modular transit system enables operators to significantly reduce operational costs and represents a worthwhile urban internal transportation system to replace traditional public transit and delivery systems. To achieve more efficient MAU allocation on the Space-Time Network, operators should encourage close cross-functional collaboration between MAU scheduling and demand forecasting teams, supported by the robust heuristic methods for large-scale instances proposed in this study, ensuring that FRT and DRT services mutually reinforce each other. The insights and methods proposed in this research are also broadly applicable to similar vehicle allocation scenarios combining fixed-route and demand-responsive services, providing a more flexible and sustainable methodological platform for the emerging urban transportation field.

8.1. Discussion and future work

Although the proposed model and results provide encouraging evidence that the passenger and freight co-transportation modular transit system can improve operational efficiency and cost savings, there are still some limitations. First, the current system only supports passenger transportation on fixed lines. Based on the results, demand-responsive freight transportation with time windows can operate efficiently, and passenger demand with precise timing could also be explored for transportation between nodes beyond fixed lines. Additionally, more detailed information on real-time passenger behavior, freight uncertainty, and geographically specific traffic patterns could enhance the accuracy of quantitative estimates and guide more targeted management strategies. Furthermore, while the customized ALNS algorithm proposed in this study has been validated to compute large-scale cases within an acceptable time frame, it still requires significant time for ultra-large-scale networks, making it difficult to meet the real-time requirements of operators. To address these limitations, the following directions for future research are proposed.

A valuable extension would be to expand passenger transportation to nodes beyond fixed lines, enabling a more comprehensive demand-responsive transportation (DRT) mode. However, transporting individual passengers incurs high costs for both operators and passengers, so the clustering of passenger demand must be considered. Future research could introduce passenger demand clustering mechanisms, such as through clustering algorithms (e.g., K-means) or spatiotemporal heatmap analysis of passengers' origin-destination patterns, allowing MAUs to dynamically pick up and drop off passengers between non-fixed-line nodes. This would not only enhance the system's flexibility but also prevent efficiency losses due to excessive dispersion. Meanwhile, such an extension may require redesigning the Space-Time Network, incorporating more dynamic arcs to support door-to-door passenger services.

Another direction is to integrate uncertainty into the model to address the real-time variability of passengers and freight. The current deterministic assumptions may lead to estimation biases, failing to account for factors such as random passenger arrivals or cancellations, unexpected freight delays, and traffic variations caused by urban congestion or weather conditions. To tackle these uncertainties, future research could employ Robust Optimization or Stochastic Programming methods, such as Distributionally Robust Optimization (DRO), to model the uncertain distribution of demand. By leveraging historical data or Monte Carlo simulations to generate scenario trees, MAU routes can be optimized to minimize operational costs under worst-case scenarios. This would make the co-transportation modular system more adaptive.

Finally, to address the computational challenges of ultra-large-scale networks, Column Generation and Row Generation methods can be explored to enhance solution performance. The generation approach decomposes the main problem into path generation subproblems, dynamically adding efficient paths (e.g., solving a shortest path variant via the pricing problem) to progressively approach the optimal solution with faster computation speed, making it particularly suitable for this Path-based MAU Routing Problem.

In summary, the expansion of passenger transport to DRT, the incorporation of uncertainty factors, and the integration of advanced decomposition methods demonstrate that the passenger and freight co-transportation modular transit system can indeed achieve further improvements in operational performance. Although many directions remain to be explored in future research, this study provides practical methodological guidance in both modeling and algorithms. In an urban transportation industry that prioritizes speed, flexibility, and cost competitiveness, the seamless integration of public transport and logistics services will be a key pathway to achieving sustained growth and success.

References

Bendibao (2025). *Shanghai Bus Query*. Accessed: 2025-07-26.

EV Connect (2025). *EV Charging Costs at Public Stations*. Accessed: 2025-07-22.

Hassold, S. and Ceder, A. ((2014). "Public Transport Vehicle Scheduling Featuring Multiple Vehicle Types". In: *Transportation Research Part B: Methodological* **67** 129–143, 129–143.

Hatzenbühler, J. et al. (2023). "Modular Vehicle Routing for Combined Passenger and Freight Transport". In: *Transportation Research Part A: Policy and Practice* **173** 103688, 103688.

Hatzenbühler, J. et al. (2024). "Multi-Purpose Pickup and Delivery Problem for Combined Passenger and Freight Transport". In: *Transportation* 1–32, 1–32.

Holguín-Veras, J., Amaral, J. C., and Rivera-Gonzalez, C. (2024). "Using GPS Data to Assess the Effectiveness of Freight Demand Management Strategies". In: *Transportation Research Procedia* **79** 21–28, 21–28.

Li, L., Negenborn, R. R., and De Schutter, B. (2013). "A General Framework for Modeling Intermodal Transport Networks". In: *Proceedings of the 10th IEEE International Conference on Networking, Sensing and Control (ICNSC)*. IEEE, pp. 579–585.

Li, S. et al. (2023). "Optimizing a Shared Freight and Passenger High-Speed Railway System: A Multi-Commodity Flow Formulation with Benders Decomposition Solution Approach". In: *Transportation Research Part B: Methodological* **172** 1–31, 1–31.

Lin, J., Nie, Y. (, and Kawamura, K. (2022). "An Autonomous Modular Mobility Paradigm". In: *IEEE Intelligent Transportation Systems Magazine* **15** (1) 378–386, 378–386.

Lin, J. and Zhang, F. (2024). "Modular Vehicle-Based Transit System for Passenger and Freight Co-Modal Transportation". In: *Transportation Research Part C: Emerging Technologies* **160** 104545, 104545.

Liu, X., Qu, X., and Ma, X. (2021). "Improving Flex-Route Transit Services with Modular Autonomous Vehicles". In: *Transportation Research Part E: Logistics and Transportation Review* **149** 102331, 102331.

Machado, B., Pimentel, C., and Sousa, A. de (2023). "Integration Planning of Freight Deliveries into Passenger Bus Networks: Exact and Heuristic Algorithms". In: *Transportation Research Part A: Policy and Practice* **171** 103645, 103645.

Mason, W. F. (1967). "Air Freight". In: *Financial Analysts Journal* **23** (5) 49–56, 49–56.

NEXT Modular Vehicles (2024). *Smart Urban Transportation*. Modular electric vehicles for sustainable urban mobility. NEXT Modular Vehicles. URL: `https://www.next-future-mobility.com` (visited on 07/17/2024).

Ropke, S. and Pisinger, D. (2006). "An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows". In: *Transportation Science* **40** (4) 455–472, 455–472.

Shanghai Municipal Human Resources and Social Security Bureau (2025). *Shanghai Human Resources and Social Security Bureau Website*. Accessed: 2025-07-26.

Shaw, P. (1997). *A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems*. Tech. rep. Glasgow, Scotland, UK: APES Group, Dept of Computer Science, University of Strathclyde, p. 46.

Shi, X. and Li, X. (2021). "Operations Design of Modular Vehicles on an Oversaturated Corridor with First-in, First-out Passenger Queueing". In: *Transportation Science* **55** (5) 1187–1205, 1187–1205.

Solar Impulse Foundation (2025). *Solutions Explorer*. Accessed: 2025-07-22.

Tang, C. et al. (2024). "Optimisation of a New Hybrid Transit Service with Modular Autonomous Vehicles". In: *Transportmetrica A: Transport Science* **20** (2) 2165424, 2165424.

Xia, D., Ma, J., and Sharif Azadeh, S. (2024a). "Integrated Timetabling and Vehicle Scheduling of an Intermodal Urban Transit Network: A Distributionally Robust Optimization Approach". In: *Transportation Research Part C: Emerging Technologies* **162** 104610, 104610.

Xia, D., Ma, J., and Sharif Azadeh, S. (2024b). "Integrated Timetabling, Vehicle Scheduling, and Dynamic Capacity Allocation of Modular Autonomous Vehicles under Demand Uncertainty". arXiv:2410.16409.

Xia, D., Ma, J., Sharif Azadeh, S., and Zhang, W. (2023). “Data-Driven Distributionally Robust Timetabling and Dynamic-Capacity Allocation for Automated Bus Systems with Modular Vehicles”. In: *Transportation Research Part C: Emerging Technologies* **155** 104314, 104314.

Yang, Y., Xie, X., and Wang, Z. (2024). “Optimization of High-Speed Rail Express Transportation Scheduling Based on Spatio-Temporal-State Network Modeling”. In: *Applied Sciences* **14** (22) 10456, 10456.

Zeng, Z. and Qu, X. (2022). “Optimization of Electric Bus Scheduling for Mixed Passenger and Freight Flow in an Urban-Rural Transit System”. In: *IEEE Transactions on Intelligent Transportation Systems* **24** (1) 1288–1298, 1288–1298.

Zhou, C. et al. (2016). “Predicting the Passenger Demand on Bus Services for Mobile Users”. In: *Pervasive and Mobile Computing* **25** 48–66, 48–66.

Zhu, S. et al. (2023). “Co-Modality in City Logistics: Sounds Good, But How?” In: *Transportation Research Part A: Policy and Practice* **168** 103578, 103578.

A

Detailed MAU path distribution of case study for co-transportation system

In the case study from 8:00 to 9:30 in the morning in Changning District, Shanghai, China, by using passenger and freight co-transportation system, a total of 30 MAUs are used. Table B.1 shows the routes assigned to all MAUs:

Table A.1: MAU path distribution of case study

Vehicle ID	Path
k_0	$d_0(8:00) > s_0(8:04) > s_1(8:08) > s_2(8:15) > s_7(8:19) > s_8(8:25) > s_9(8:29) > s_{10}(8:33) > d_3(8:38)$
k_1	$d_1(8:00) > s_5(8:05) > s_4(8:12) > s_3(8:19) > s_2(8:25) > s_7(8:30) > s_8(8:35) > s_{14}(8:39) > s_{13}(8:45) > s_0(8:52) > s_1(8:56) > s_2(9:00) > s_3(9:03) > s_{12}(9:06) > s_{11}(9:08) > s_9(9:11) > s_8(9:15) > s_7(9:18) > s_6(9:21) > d_2(9:25)$
k_2	$d_2(8:00) > s_6(8:07) > s_7(8:11) > s_8(8:15) > s_9(8:20) > s_{10}(8:27) > d_3(8:31)$
k_3	$d_3(8:00) > s_{10}(8:04) > s_9(8:11) > s_8(8:16) > s_7(8:23) > s_2(8:30) > s_1(8:36) > s_0(8:43) > d_0(8:48)$
k_4	$d_0(8:15) > s_0(8:19) > s_1(8:26) > s_2(8:32) > s_7(8:36) > s_8(8:40) > s_9(8:47) > s_{10}(8:51) > d_3(8:56)$
k_5	$d_1(8:15) > s_5(8:20) > s_4(8:26) > s_3(8:33) > s_2(8:38) > s_7(8:44) > s_8(8:48) > s_9(8:55) > s_{10}(8:59) > d_3(9:04)$
k_6	$d_1(8:15) > s_5(8:22) > s_4(8:28) > s_3(8:35) > s_2(8:41) > s_1(8:46) > s_0(8:49) > d_0(8:53)$
k_7	$d_2(8:15) > s_6(8:20) > s_7(8:25) > s_2(8:30) > s_3(8:35) > s_4(8:39) > s_5(8:45) > d_1(8:50)$
k_8	$d_3(8:15) > s_{10}(8:21) > s_9(8:27) > s_8(8:31) > s_7(8:37) > s_6(8:44) > d_2(8:49)$
k_9	$d_0(8:30) > s_0(8:35) > s_1(8:41) > s_2(8:48) > s_3(8:54) > s_4(8:59) > s_5(9:03) > d_1(9:08)$
k_{10}	$d_1(8:30) > s_5(8:35) > s_4(8:41) > s_3(8:47) > s_2(8:54) > s_1(8:56) > s_0(9:01) > d_0(9:04)$
k_{11}	$d_2(8:30) > s_6(8:37) > s_7(8:43) > s_8(8:48) > s_9(8:53) > s_{10}(8:59) > d_3(9:03)$
k_{12}	$d_3(8:30) > s_{10}(8:34) > s_9(8:38) > s_8(8:45) > s_7(8:52) > s_2(8:57) > s_1(9:02) > s_0(9:05) > d_0(9:08)$

Continued on next page

Table A.1: MAU path distribution of case study

Vehicle ID	Path
k_{13}	$d_3(8:32) > s_{10}(8:37) > s_9(8:43) > s_8(8:49) > s_7(8:55) > s_6(9:00) > d_2(9:04)$
k_{14}	$d_0(8:35) > s_0(8:40) > s_1(8:47) > s_2(8:53) > s_7(8:58) > s_6(9:02) > d_2(9:07)$
k_{15}	$d_3(8:38) > s_{10}(8:42) > s_9(8:48) > s_8(8:54) > s_7(9:01) > s_2(9:05) > s_1(9:08) > s_0(9:11) > d_0(9:15)$
k_{16}	$d_3(8:42) > s_{10}(8:47) > s_9(8:52) > s_8(8:59) > s_7(9:03) > s_2(9:06) > s_3(9:10) > s_4(9:13) > s_5(9:17) > d_1(9:21)$
k_{17}	$d_0(8:45) > s_0(8:49) > s_1(8:53) > s_2(8:58) > s_3(9:03) > s_4(9:07) > s_5(9:11) > d_1(9:15)$
k_{18}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{19}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{20}	$d_3(8:45) > s_{10}(8:52) > s_9(8:56) > s_8(9:03) > s_7(9:08) > s_6(9:10) > d_2(9:13)$
k_{21}	$d_0(8:48) > s_0(8:53) > s_1(8:59) > s_2(9:05) > s_7(9:09) > s_6(9:13) > d_2(9:16)$
k_{22}	$d_3(8:48) > s_{10}(8:53) > s_9(8:59) > s_8(9:04) > s_7(9:08) > s_2(9:12) > s_1(9:15) > s_0(9:19) > d_0(9:23)$
k_{23}	$d_2(8:50) > s_6(8:54) > s_7(8:58) > s_8(9:03) > s_9(9:07) > s_{10}(9:11) > d_3(9:15)$
k_{24}	$d_0(8:50) > s_0(8:54) > s_1(8:58) > s_2(9:03) > s_3(9:07) > s_4(9:11) > s_5(9:15) > d_1(9:19)$
k_{25}	$d_0(8:55) > s_0(8:58) > s_{13}(9:02) > s_0(9:05) > s_1(9:09) > s_2(9:11) > s_3(9:15) > s_4(9:18) > s_5(9:21) > d_1(9:24)$
k_{26}	$d_0(8:55) > s_0(8:59) > s_1(9:03) > s_2(9:07) > s_3(9:11) > s_4(9:15) > s_5(9:19) > d_1(9:23)$
k_{27}	$d_1(9:00) > s_5(9:06) > s_4(9:10) > s_3(9:15) > s_2(9:19) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{28}	$d_2(9:00) > s_6(9:07) > s_7(9:11) > s_8(9:15) > s_9(9:18) > s_{10}(9:23) > d_3(9:26)$
k_{29}	$d_3(9:00) > s_9(9:05) > s_{11}(9:09) > s_8(9:13) > s_{14}(9:16) > s_1(9:19) > s_0(9:22) > d_0(9:25)$

B

Detailed MAU path distribution of case study for separate transportation system

In the case study from 8:00 to 9:30 in the morning in Changning District, Shanghai, China, by using passenger and freight separate system, a total of 30 MAUs are used. Table B.1 shows the routes assigned to all MAUs:

Table B.1: MAU path distribution for passenger and freight demand

Vehicle ID	Path
MAU for passenger demand:	
k_0	$d_0(8:00) > s_0(8:04) > s_1(8:08) > s_2(8:15) > s_7(8:19) > s_8(8:25) > s_9(8:29) > s_{10}(8:33) > d_3(8:38)$
k_1	$d_1(8:00) > s_5(8:05) > s_4(8:12) > s_3(8:19) > s_2(8:25) > s_7(8:31) > s_6(8:36) > d_2(8:41)$
k_2	$d_2(8:00) > s_6(8:07) > s_7(8:11) > s_8(8:15) > s_9(8:20) > s_{10}(8:27) > d_3(8:31)$
k_3	$d_3(8:00) > s_{10}(8:04) > s_9(8:11) > s_8(8:16) > s_7(8:23) > s_2(8:30) > s_1(8:36) > s_0(8:43) > d_0(8:48)$
k_4	$d_0(8:15) > s_0(8:19) > s_1(8:26) > s_2(8:32) > s_7(8:36) > s_8(8:40) > s_9(8:47) > s_{10}(8:51) > d_3(8:56)$
k_5	$d_1(8:15) > s_5(8:20) > s_4(8:26) > s_3(8:33) > s_2(8:38) > s_7(8:44) > s_8(8:48) > s_9(8:55) > s_{10}(8:59) > d_3(9:04)$
k_6	$d_1(8:15) > s_5(8:22) > s_4(8:28) > s_3(8:35) > s_2(8:41) > s_1(8:46) > s_0(8:49) > d_0(8:53)$
k_7	$d_2(8:15) > s_6(8:20) > s_7(8:25) > s_2(8:30) > s_3(8:35) > s_4(8:39) > s_5(8:45) > d_1(8:50)$
k_8	$d_3(8:15) > s_{10}(8:21) > s_9(8:27) > s_8(8:31) > s_7(8:37) > s_6(8:44) > d_2(8:49)$
k_9	$d_0(8:30) > s_0(8:35) > s_1(8:41) > s_2(8:48) > s_3(8:54) > s_4(8:59) > s_5(9:03) > d_1(9:08)$
k_{10}	$d_1(8:30) > s_5(8:35) > s_4(8:41) > s_3(8:47) > s_2(8:54) > s_1(8:56) > s_0(9:01) > d_0(9:04)$
k_{11}	$d_2(8:30) > s_6(8:37) > s_7(8:43) > s_8(8:48) > s_9(8:53) > s_{10}(8:59) > d_3(9:03)$

Continued on next page

Table B.1: MAU path distribution for passenger and freight demand

Vehicle ID	Path
k_{12}	$d_3(8:30) > s_{10}(8:34) > s_9(8:38) > s_8(8:45) > s_7(8:52) > s_2(8:57) > s_1(9:02) > s_0(9:05) > d_0(9:08)$
k_{13}	$d_3(8:38) > s_{10}(8:42) > s_9(8:48) > s_8(8:54) > s_7(9:01) > s_2(9:05) > s_1(9:08) > s_0(9:11) > d_0(9:15)$
k_{14}	$d_3(8:42) > s_{10}(8:47) > s_9(8:52) > s_8(8:59) > s_7(9:03) > s_2(9:06) > s_3(9:10) > s_4(9:13) > s_5(9:17) > d_1(9:21)$
k_{15}	$d_0(8:45) > s_0(8:49) > s_1(8:53) > s_2(8:58) > s_3(9:03) > s_4(9:07) > s_5(9:11) > d_1(9:15)$
k_{16}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{17}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{18}	$d_3(8:45) > s_{10}(8:52) > s_9(8:56) > s_8(9:03) > s_7(9:08) > s_6(9:10) > d_2(9:13)$
k_{19}	$d_0(8:48) > s_0(8:53) > s_1(8:59) > s_2(9:05) > s_7(9:09) > s_6(9:13) > d_2(9:16)$
k_{20}	$d_3(8:48) > s_{10}(8:53) > s_9(8:59) > s_8(9:04) > s_7(9:08) > s_2(9:12) > s_1(9:15) > s_0(9:19) > d_0(9:23)$
k_{21}	$d_2(8:50) > s_6(8:54) > s_7(8:58) > s_8(9:03) > s_9(9:07) > s_{10}(9:11) > d_3(9:15)$
k_{22}	$d_0(8:50) > s_0(8:54) > s_1(8:58) > s_2(9:03) > s_3(9:07) > s_4(9:11) > s_5(9:15) > d_1(9:19)$
k_{23}	$d_0(8:55) > s_0(8:58) > s_1(9:03) > s_2(9:06) > s_3(9:10) > s_4(9:14) > s_5(9:18) > d_1(9:21)$
k_{24}	$d_0(8:55) > s_0(8:59) > s_1(9:03) > s_2(9:07) > s_3(9:11) > s_4(9:15) > s_5(9:19) > d_1(9:23)$
k_{25}	$d_1(9:00) > s_5(9:06) > s_4(9:10) > s_3(9:15) > s_2(9:19) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{26}	$d_2(9:00) > s_6(9:07) > s_7(9:11) > s_8(9:15) > s_9(9:18) > s_{10}(9:23) > d_3(9:26)$
k_{27}	$d_3(9:00) > s_9(9:05) > s_{11}(9:09) > s_8(9:13) > s_{14}(9:16) > s_1(9:19) > s_0(9:22) > d_0(9:25)$
MAU for freight demand:	
k_{28}	$d_0(8:55) > s_0(8:58) > s_{13}(9:02) > s_0(9:05) > s_1(9:09) > s_2(9:11) > s_3(9:15) > s_4(9:18) > s_5(9:21) > d_1(9:24)$
k_{29}	$d_1(8:58) > s_5(9:03) > s_4(9:04) > s_{12}(9:06) > s_{11}(9:08) > s_9(9:11) > s_8(9:15) > s_7(9:18) > s_6(9:21) > d_2(9:25)$
k_{30}	$d_2(9:00) > s_6(9:04) > s_7(9:07) > s_8(9:10) > s_{12}(9:15) > s_3(9:19) > s_4(9:22) > s_5(9:25) > d_1(9:28)$
k_{31}	$d_1(9:00) > s_5(9:06) > s_4(9:10) > s_3(9:15) > s_2(9:19) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{32}	$d_2(9:00) > s_6(9:07) > s_7(9:11) > s_8(9:15) > s_9(9:18) > s_{10}(9:23) > d_3(9:26)$
k_{33}	$d_3(9:00) > s_9(9:05) > s_{11}(9:09) > s_8(9:13) > s_{14}(9:16) > s_1(9:19) > s_0(9:22) > d_0(9:25)$
k_{34}	$d_3(9:02) > s_{10}(9:06) > s_9(9:10) > s_8(9:13) > s_7(9:17) > s_6(9:20) > d_2(9:24)$
k_{35}	$d_3(9:02) > s_{10}(9:06) > s_9(9:10) > s_8(9:13) > s_7(9:17) > s_2(9:20) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{36}	$d_2(9:03) > s_6(9:06) > s_7(9:09) > s_8(9:12) > s_9(9:15) > s_{10}(9:20) > d_3(9:23)$

Continued on next page

Table B.1: MAU path distribution for passenger and freight demand

Vehicle ID	Path
k_{37}	$d_0(9:05) > s_0(9:08) > s_1(9:11) > s_{13}(9:14) > s_{14}(9:18) > s_7(9:21) > s_6(9:24) > d_2(9:27)$

C

Journal format

Optimization of passenger and freight co-transportation in modular transit systems

Chenwei Peng, Dongyang Xia, Yahan Lu, Yousef Maknoon, Sh. Sharif Azadeh
Department of Transport & Planning, Delft University of Technology, Netherlands

The daily fluctuations in urban passenger demand lead to space wastage in public transportation vehicles, increasing unnecessary operational costs. Meanwhile, with the expansion of e-commerce, urban freight demand continues to grow, making the use of idle public transportation space for freight transport an ideal approach. This study proposes an optimization framework for integrated passenger-freight co-transportation using Modular Autonomous Vehicles (MAVs), formulating a Mixed-Integer Quadratically Constrained Programming (MIQCP) Path-based model based on a Space-Time Network to address the Modular Autonomous Unit (MAU) Routing Problem. The model integrates Fixed-Route Transit (FRT) and Demand-Responsive Transit (DRT), allowing MAUs to dynamically couple/decouple across different routes to meet the spatio-temporal demands of passengers and freight, with the objective of minimizing total operational costs. To tackle the computational complexity of large-scale instances, a customized Adaptive Large Neighborhood Search (ALNS) algorithm is designed, incorporating two initial solution generation methods (GUROBI and Greedy heuristic) and iteratively optimizing solutions through destroy and repair operations. A real-world case study based on the Shanghai bus network validates the effectiveness of the proposed approach. The results demonstrate that the MAU co-transportation system can effectively utilize vehicle compartment space to simultaneously transport passengers and freight, significantly reducing empty load rates, leading to a substantial reduction in operating costs. Without using the co-transportation mode, the number of MAUs used would increase significantly, accompanied by a 4.1% cost increase. Compared to the traditional combination of public transit and delivery vans, costs are reduced by 502.1%. This co-transportation modular transit system, with its unique flexibility, can provide efficient and low-cost transportation services for both passengers and freight within cities.

Key words: Modular Autonomous Vehicle, Co-transportation modular transit system, MAU Routing Problem, Adaptive Large Neighborhood Search

1. Introduction

In recent years, urban freight demand has continued to grow alongside the expansion of e-commerce. Meanwhile, the daily fluctuations in urban passenger demand make it challenging to balance public transport operating costs and passenger waiting times (Shi and Li 2021). During off-peak hours when passenger demand is low, maintaining the same bus arrival frequency as during peak periods leads to low space utilization and increased operating costs. However, reducing bus frequencies would result in longer waiting times for passengers, thereby decreasing service satisfaction.

Machado, Pimentel, and de Sousa (2023) proposed retrofitting some buses so that part of their interior space is used for passenger transport while the remaining space is allocated for freight transportation. However, under uncertain demand conditions, this modification still struggles to serve as an optimal solution. Modular Autonomous Vehicle (MAV) can reduce capacity redundancy since the capacity of a single Modular Autonomous Unit (MAU) is smaller than that of a traditional bus. It also has the function of transporting both passengers and freight in one carriage, and can better respond to changes in demand.

NExT Company has currently designed a mature MAV system (NExT Modular Vehicles 2024), as shown in Figure 1, which can achieve coupling/decoupling while in motion and can open front and rear carriage doors when coupled. This function ensures that passengers can freely move between multiple coupled MAUs, making transfers between different routes more convenient. Additionally, NExT Company has proposed a feature that allows freight to be transferred to different MAUs through automated machinery, enabling more flexible freight pick-up and delivery across different routes by moving freight between modules when MAUs are coupled. One of the important advantage of NExT Company's MAV over traditional public transportation is that the autonomous driving feature reduces labor costs, with only energy costs during operation, and provides greater flexibility. It can precisely fulfill passenger demands when they arise, controlling passenger waiting time at stations. Meanwhile, when there is spare capacity in the carriage, it can meet freight demands within designated time windows.

Figure 1 **NExT Company's MAV**

As a product of recent advancements in autonomous driving technology, research on MAUs is still in its early stages, with most studies focusing solely on passenger demand and operating on a single bus line. For integrated passenger-freight transport systems, current research primarily

focuses on transportation along fixed lines (Lin and Zhang 2024) or treats MAUs as taxis capable of transporting freight (Hatzenbühler et al. 2024), starting from a depot to execute a series of pre-booked tasks before returning to the origin. This study developed a system model for passenger and freight co-transportation to address the research gap by completing the routing allocation for each MAU on the network. And a customized Adaptive Large Neighborhood Search (ALNS) algorithm was utilized to obtain the optimal feasible solution more quickly and accurately.

By analyzing passenger behavior, daily passenger demand at stations along fixed bus lines can be accurately estimated for specific time periods (Zhou et al. 2016), and since that the bus timetable is known, the vast majority of passengers are assumed to arrive at stations according to the timetable, reducing their waiting time. Additionally, pick-up and delivery demands for freight can be predicted in advance using GPS positioning technology (Holguín-Veras, Amaral, and Rivera-Gonzalez 2024). The MAU in this study is designed to function both as a bus, transporting passengers and freight along fixed bus lines, and as a delivery vehicle, transporting freight between pick-up and delivery stops outside the lines. Additionally, each MAU is capable of operating across multiple lines.

This study models the MAV Routing Problem as a Mixed-Integer Quadratically Constrained Programming (MIQCP) model based on a Space-Time Network, integrating autonomous units and paths. The Space-Time Network includes depots and stops at all discrete timestamps, connected by Travel arcs, Delivery arcs, Transfer arcs, and Waiting arcs to form paths. The model aims to find the minimum cost required for MAUs to fulfill all transportation demands. The results display the transportation paths for each MAU and the allocation of each transportation demand. A customized Adaptive Large Neighborhood Search (ALNS) algorithm is employed to compute the results, with its quality compared against exact solutions obtained from the GUROBI solver in small-scale tests. Two initial solution methods are used: one selects a small number of demand-containing paths for GUROBI computation, and the other applies a Greedy heuristic method to quickly assign all demands to MAUs. The ALNS iteratively destroys and reconstructs parts of the current solution to find better solutions. The destruction and repair processes are divided into two levels: Path-level, involving large-scale deletion and reconstruction of entire paths, and Arc-level, involving deconstruction and reconstruction of selected paths. The algorithm is customized with multiple operators tailored to the MAV Routing Problem, including destroy operators (Path-level: Random Destroy, Worst Cost Destroy, Demand-based Destroy; Arc-level: Arc Removal Destroy, Arc Search Destroy) and repair operators (Path-level: Random Repair, Greedy Repair, Utility Maximization Repair; Arc-level: Arc Insertion Repair, Chain Repair, Hybrid Repair), with operator weights dynamically adjusted using Roulette Wheel Selection. The ALNS uses Simulated Annealing (SA) as the acceptance criterion, accepting not only improved solutions but also worse solutions with a certain probability to escape local optima. By dynamically adjusting the destruction rate

and operator scoring mechanism, this ALNS achieves a balance between local optimization depth and global search breadth.

From an academic perspective, this study contributes to the transportation field in two main ways. First, it expands the body of knowledge in this area by developing an optimization model for passenger and freight co-transportation modular transit systems on a Space-Time Network. Second, it explores the scalability of the model in large-scale systems by proposing an efficient Adaptive Large Neighborhood Search (ALNS) algorithm tailored to the problem's inherent complexity. Additionally, it provides new methodological contributions for the integrated utilization of internal space in public transportation systems.

From a societal perspective, the results of this study demonstrate that using MAUs for co-transportation can significantly reduce the operational costs of transporting passengers and freight within cities, potentially becoming a fundamental service for future passenger and freight transport. This system can maximize the utilization of carriage space while reducing the average waiting time for passengers at stations. Furthermore, the effective reduction in the number of vehicles used leads to a significant decrease in electricity consumption, promoting more sustainable urban communities.

The structure of the remainder of this thesis is as follows. Section 2 discusses the research questions of this study and outlines the approach to addressing them. Section 3 reviews related literature and summarizes the contributions of this study. Section 4 provides the problem description. In Section 5, a Mixed-Integer Quadratically Constrained Programming (MIQCP) model for the MAV Routing Problem is established. Then, in Section 6, a customized ALNS algorithm is designed, and its operational process is detailed. Section 7 presents the computational results, comparing the accuracy and computational time of different methods, and includes a case study based on real-world bus lines and pick-up and delivery stops in Shanghai. Section 8 offers conclusions and directions for future research.

2. Research questions

To establish a passenger and freight co-transportation system using MAUs within cities, the following core questions need to be answered:

Is passenger and freight co-transportation in modular transit systems worthy of large-scale application within cities?

The main research question is answered by the following sub-questions. Table 1 shows these sub-questions and provides solutions:

Table 1 Sub-questions for research and corresponding solutions

Sub-questions	Methods
1) What aspects does the network of this study consider?	Space-Time Network
2) What kind of mathematical model is best suited to solve this MAU Routing Problem?	Path-based Mixed Integer Linear Programming
3) What methods are used to solve the optimization model?	GUROBI and Heuristic Method (Adaptive Large Neighborhood Search)
4) How do solution methods perform at different scales?	Numerical experiments based on simulated data
5) Compared with separate modular transit system and traditional transportation system, how big is the advantage of MAV?	Numerical experiments based on real data

This research examines the necessity and feasibility of establishing a passenger and freight co-transportation modular transit system within cities to improve transportation efficiency and cost. Due to the complexity of the problem, large-scale solvers often struggle to find high-quality solutions within an acceptable time frame. Heuristic algorithms offer the potential for greater computational efficiency. In general, these research questions aim to demonstrate the significant potential of this system in terms of flexibility, space utilization, and sustainability, making it a worthy candidate for development as a new type of transportation system.

3. Literature review

At present, the research on the Passenger-Freight intermodal transport problem of Modular Autonomous Vehicles (MAV) is in its infancy, and there are still many gaps to be explored. Therefore, this section mainly reviews the construction of Space-Time networks in the Passenger-Freight intermodal transport problem of other transportation modes and the current research status of MAVs.

3.1. Passenger-Freight intermodal transport based on the Space-Time network

Passenger-Freight intermodal transport has been used for decades for long-haul transport, such as air transport (Mason 1967) and rail transport (Yang, Xie, and Wang 2024). The modeling approach of constructing a Space-Time network to optimize the shared capacity of the two services can integrate different modes of transportation, support the dynamic changes in passenger and cargo demand, and flexibly adjust transportation routes and schedules (Li, Negenborn, and De Schutter 2013). In order to solve the problem of integrating passenger and freight services on the same railway network and maximize the transportation efficiency and cost-effectiveness of the network, Li et al. (2023) imposed constraints on train, station capacity and freight delays, then calculated the minimum service and routing costs. Only in recent years has short-haul transport, especially intermodal transport within cities, begun to gain popularity (Zhu et al. 2023).

Passenger-Freight intermodal transport by bus is one of the most realistic forms in the cities and is most similar to the Modular Autonomous Vehicles (MAVs) studied in this thesis. Zeng and Qu (2022) built the network taking into account customer pick-up time windows, loading/unloading service duration, and the power supply needs of electric buses. To minimize operating costs, a Mixed Integer Linear Programming model is developed to meet the bus schedule for passenger travel needs, cargo delivery needs and charging requests. Machado, Pimentel, and de Sousa (2023) further considered the uncertainty of cargo demand and establishes a model through demand scenarios in different time and space. The buses in the current study all need to be modified, with fixed passenger and cargo capacity, and there will still be problems with redundant passenger or cargo compartments in reality. The dynamic demand of passengers for direct access cannot be met by buses, and other modes of transportation are required for the last mile (Machado, Pimentel, and de Sousa 2023). The small capacity of each unit and the coupling/decoupling characteristics of MAVs can effectively make up for the shortcomings of buses in urban intermodal transport (Shi and Li 2021).

3.2. Optimization methods related to MAU

In recent years, with the development of autonomous driving technology, the emergence of modular vehicles has opened up new directions for the exploration of public transportation networks. When

establishing the optimal timetable for each bus route based on the minimum cost network flow model, Hassold and Ceder (2014) proposed to use multiple small vehicles to replace the original planned models. The results showed that this method can significantly reduce operating costs and improve service quality. Modular autonomous vehicles (MAVs) can change the formation by changing the number of composed MAUs according to the time-varying passenger demand at different stations (Shi and Li 2021), so more research is currently focused on establishing a cost-controlled public transportation network that serves passengers with travel needs.

Liu, Qu, and Ma (2021) proposed that each MAU can freely visit customers outside the check-point, and deal with the first-mile and last-mile passenger pick-up problem by determining the best route and scheduling strategy. A Mixed Integer Linear Programming that not only focuses on vehicle operating costs, passenger waiting and in-vehicle costs, but also adds penalties for not responding to passenger needs is designed. To solve this problem, they used a customized dynamic programming with valid cuts in the first stage to effectively arrange the dynamic time and route of vehicles, and proposed an effective and fast heuristic method in the second stage to solve the dynamic allocation and path problem, and obtained a plan for real-time allocation and scheduling of vehicles in a region. Tang et al. (2024) considered that the areas between stations should be distinguished, and the service areas and routes of the day are selected according to the passengers' online reservations one day in advance, which will cause some passengers to walk to another nearby station that will be served. The MAU will separate from the MAV fleet at the starting point of the selected section and reconnect with the fleet at the end of these sections. They developed an Optimization model to determine the deviated sections and the number of MAUs to be separated, as well as the corresponding schedule, to minimize the waiting time of passengers and the total cost of operator operation. The results are obtained using the DICOPT solver and show that the number of door-to-door passengers would affect the total cost. The more such passengers there are, the higher the waiting time and walking time cost will be, while the in-vehicle time and vehicle operation cost will be reduced.

Xia et al. (2023) focused on the uncertainty of passengers' time-dependent demand, and seeked the lowest-cost and best robust MAV time-varying and station-varying capacity allocation plan and the corresponding schedule. Due to the random nature of passenger arrival rate, the number of MAVs will change over time in a trip, and different MAVs will have different numbers of MAUs at the same station. A model of the Timetable and Dynamic Capacity Allocation (TT-DCA) problem is proposed and extended by the Data-Driven Distributionally Robust Optimization (DRO) method to consider the uncertainty of passengers' arrival time. The Integer L-shaped (IL) method can better solve this problem. The results show that the allocation strategy of time-varying and station-varying capacity can lead in both maximum and average vehicle congestion levels, indicating that the waste

of space in the vehicle can be effectively reduced. According to Xia, Ma, and Sharif Azadeh (2024a), the passenger demand for direct access was added to the fixed route, and the MU at any station can be separated from the MAV to complete the DRT service, and after completing the service, it can continue to couple with the MAV that coincides with the time. The Mixed Integer Linear Programming model is established to generate a globally optimal co-mobility schedule and service route (minimizing the weighted sum of passenger and operating costs). By using a customized Adaptive Large Neighborhood Search (ALNS) algorithm combined with the GUROBI solver, it is found that the OT-FC-DRT strategy makes the number of operating vehicles and operating costs higher than the OT-FC strategy, but reduces the waiting time cost and average in-vehicle time of passengers. If the cost weight is adjusted to favor the operator, the DRT service may be sacrificed. Furthermore, the dynamic constraint of passengers transferring between different routes was added by Xia, Ma, and Sharif Azadeh (2024b), so that the model can realize the function of finding a scheduling scheme and schedule that utilizes fewer MUs while establishing multi-line circulation. Based on the Integer L-shaped (IL) method, the Rolling Horizon Framework (RH) has been proposed to address efficiency issues found in numerical experiments based on Beijing's bus network data. It works by dynamically and incrementally solving the problem to adapt to the real-time changes and uncertainties in demand. For the three modes of Fixed-capacity, Partially flexible-capacity and Completely flexible-capacity, the costs for passengers do not differ much in different cases, but the Completely flexible-capacity mode has huge advantages in terms of operating costs and the number of vehicles used. However, if more attention is paid to the interests of passengers, the operating costs will increase, but the proportion of transfers within the vehicle can be increased, making it more convenient for passengers to transfer.

The use of MAVs for passenger and freight transport is a relatively new concept. Due to the low demand of passengers during off-peak periods, there is more redundancy in vehicles, while the rise of e-shopping has increased the demand for sending and receiving goods. MAVs that can quickly connect and detach are a promising solution for a co-modal system that integrates public transportation services and last-mile logistics (Lin, Nie, and Kawamura 2022). Hatzenbühler et al. (2023) proposed two independent dedicated MAUs for passengers and freight. They modeled the Modular Multi-purpose Pickup and Delivery Problem (MMP-PDP) by considering the travel time cost, travel distance cost, fleet size cost, and the cost of unserved requests. The transportation mode only involves demand response type. Under the condition of meeting the time window constraints, the corresponding group of MAVs are arranged to depart from the depot and complete a series of tasks of picking up and dropping off passengers and goods in the shortest path before returning to the depot. Experiments show that while the operating cost is reduced by 48%, the travel time is also much shorter than that of ordinary fleets. Lin and Zhang (2024) proposed a different concept,

where the same MAU serves both passengers and cargo. The number and task allocation of MAUs can be adjusted at any station to minimize the waiting time cost of passengers and the operating cost of operators. A fixed route in both directions is established, but since the two directions share the module inventory at the same station, the planning of both directions will affect each other. In order to simplify the established Mixed Integer Programming problem, the time coordinate is shifted to focus on the time when each MAV departs from the first station. A two-stage heuristic algorithm is used to solve this problem. The first stage determines the number and the timetable of vehicles that need to be scheduled, and the second stage develops a high-quality lower bound to optimize vehicle grouping and freight allocation. They finally found that an increase in freight demand and an increase in the maximum allowed platoon length can further reduce the total cost.

The following Table 2 shows the current research status of Modular autonomous vehicles (MAVs) and the research direction of this thesis. At present, most of the research on MAVs with only passengers is exploring the service mode that combines Fixed-route transit (FRT) and Demand-responsive transit (DRT). Such an intermodal mode can better reduce the empty vehicle rate and improve passenger satisfaction. However, there are fewer studies on passenger-freight intermodal transport, and most of them are one of the two service forms of FRT (Lin and Zhang 2024) and DRT (Hatzenbühler et al. 2024). The combination of these two service forms has been found to be more effective in previous studies on only passengers, so it is necessary to explore the impact of the combination of these two service forms on the cost control of MAVs' passenger-freight i2ntermodal transport.

Pas: Passenger, Fre: Freight, FRT: Fixed-route transit, DRT: Demand-responsive transit, DP: Dynamic programming, IL: Integer L-shaped, ALNS: Adaptive large neighborhood search algorithm, RH: Rolling horizon framework

Publications	**Demand**	**Transport Mode**	**Flexibility of Vehicle Formation**	**Objective**	**Solution**
Liu, Qu, and Ma (2021)	Pas	FRT + DRT	Docked/undocked at each stop	Operating cost, Waiting time cost, Travel time, Penalty cost for unserved demand	DP + Heuristic
Xia et al. (2023)	Pas	FRT	Docked/undocked at each stop	Operating cost, Waiting time cost	IL
Tang et al. (2024)	Pas	FRT + DRT	Docked/undocked at selected stops	Vehicle ownership, Operating cost, Maintenance cost, Waiting time cost, Travel time	Solver (DICOPT)
Xia, Ma, and Sharif Azadeh (2024a)	Pas	FRT + DRT	Docked/undocked at each stop	Operating cost, Waiting time cost, Travel time	ALNS + Solver (GUROBI)
Xia, Ma, and Sharif Azadeh (2024b)	Pas	FRT + DRT (network)	Docked/undocked at each stop	Operating cost, Waiting time cost	IL + RH
Hatzenbühler et al. (2023)	Pas + Fre	DRT	Docked/undocked at terminals only	Operating cost, Vehicle ownership, Travel time, Penalty cost for unserved demand	ALNSA + Solver (CPLEX)
Lin and Zhang (2024)	Pas + Fre	FRT	Docked/undocked at each stop	Operating cost (vehicles and stations), Waiting time cost, freight handling cost	Two stage Heuristic
this thesis	Pas + Fre	FRT + DRT (network)	Docked/undocked at each stop	Operating cost	ALNS + Solver (GUROBI)

Table 2 Comparison of Various MAVs Studies

4. Problem description

This study considers the transportation services of multiple bus lines in a region within one day (24 hours). The fixed lines have starting and ending points, set as depots, denoted by $d \in D$. The remaining stops are denoted by $s \in S$. There are transfer stops between different lines, intersections of two lines. These stations are represented by different numbers s on different lines but are spatially the same. Independent of the lines, the freight pickup and delivery stops are also denoted by $s \in S$, but these stops can change daily for providing door-to-door service. Modular Autonomous Vehicles (MAVs) pick up and drop off passengers only on fixed routes, while for freight, they can automatically move within the region to respond to demand at pickup and delivery stops.

In the bus system, the distribution of passengers typically exhibits dynamic characteristics at different times and different stations. During peak hours, the number of passengers is usually significantly higher than during off-peak hours, which results in more redundancy of MAUs during off-peak periods. Therefore, utilizing the remaining vehicle capacity to transport goods during off-peak periods, while meeting passenger transportation demands, is a more effective method to improve utilization. Adopting a flexible grouping mode that combines passenger and freight transport can significantly reduce the operational cost losses caused by the idleness of MAUs while ensuring service levels. Each MAU can couple/decouple at various stations and can be used for both passenger and freight transport. Since the automatic robot can assist the freight to move between different connected MAUs, as shown in Figure 2, and passengers can also freely shuttle between MAUs, as shown in Figure 3, this study assumes that both passengers and freight can reach the correct MAU before transferring, so there will be no penalty for transfers included in the cost. To maximize the utilization of vehicle space, passengers and freights can coexist in the same MAU when operating on fixed routes. However, if an MAU needs to go to a pick-up and delivery stop, it cannot have passengers on board. This transport mode is shown in Figure 4. Considering that MAUs can have different entrance and exit stations and time windows, each MAU is denoted by $k \in K$, and has a capacity limit Q.

This thesis focuses on the MAV Routing Problem in a multi-depot system with intersecting MAV bus lines, employing flexible grouping and rerouting modes to meet both passenger and freight transportation demands. The objective of the mathematical model is to minimize the operational costs of all used MAUs. It is assumed that the schedules of MAVs, the number of MAUs, node locations, and the quantities of passenger/freight transportation demands are known. The bus timetable is known (FRT), but when a MAU is free, it can start at any time to perform the task of transporting freight and assist in completing some passenger demands (DRT). Additionally, the paths between each station are associated with pre-determined operational costs (proportional to travel time), and it is assumed that the power consumption is the same whether there are passengers

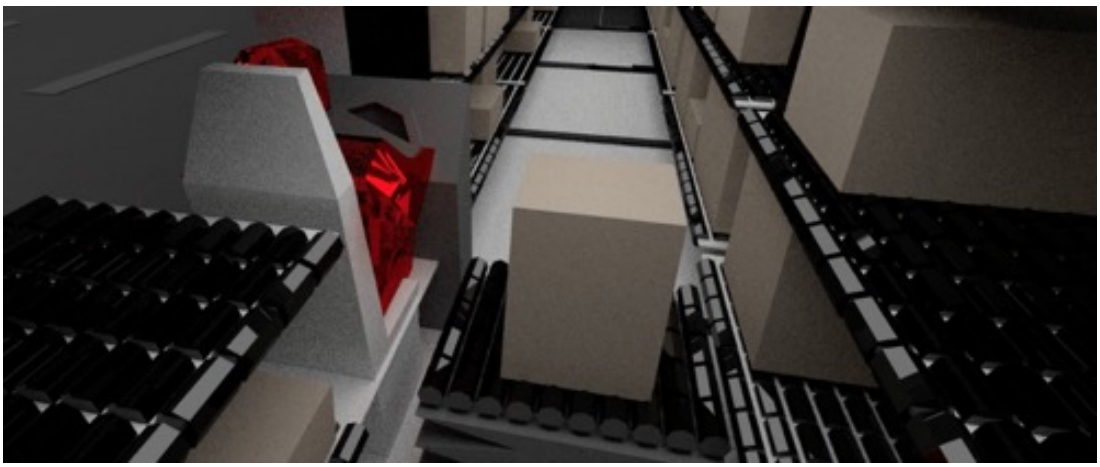

Figure 2 **Freight movement between MAUs**

Figure 3 **Passenger movement between MAUs**

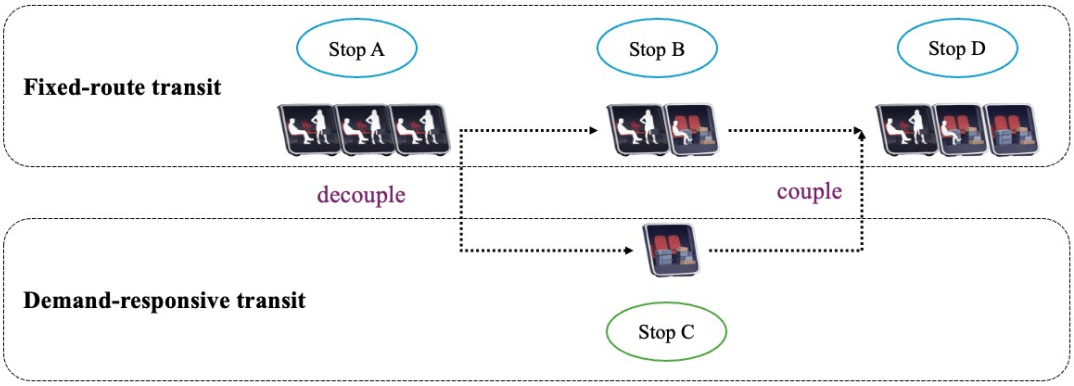

Figure 4 **MAV transport mode**

or freight in the MAU. The main reason why the cost of MAU is linked to travel time is that this study mainly considers the operating costs that operators need to pay and travel time affects

energy consumption and battery efficiency. The longer the electric bus runs, the higher the energy consumption, resulting in higher operating costs. And for the same driving distance, in order to compare the difference in cost between peak hours and off-peak hours, it is more accurate to use time as the objective. The purpose is to determine the space-time trajectories of all operating MAUs. To describe the Space-Time movements of vehicles, passengers, and freight in the bus system, the Space-Time network representation method is introduced in the modeling process.

4.1. Space-Time network

The Space-Time network can clearly capture the working status and movement trajectory of each MAU at different timestamps. In the network, MAUs with coinciding Space-Time states automatically form a column of MAVs. To construct the Space-Time network, the time range is discretized into a set of timestamps according to a time granularity, denoted as $T = \{t_0, t_1, ..., t_T\}$. The timestamp is divided into 1-minute intervals, so there are 1440 timestamps in one day. Depots and Stops are expanded into a series of Space-Time vertices at discrete timestamps. A Space-Time vertex can be represented in a unified form as (d,t) or (s,t), indicating the location of an MAU at timestamp t. The set N^R represents all Space-Time vertices. The movement of MAUs can be represented by directed arcs between Space-Time vertices. The set of Space-Time arcs is denoted by A^R, where each Space-Time arc is represented as (d,t,s,t') or (s,t,s',t') or (s,t,d,t'), describing the movement process of an MAU from the starting point to the destination. t represents the timestamp at station i, and $t' = t + t_a$, where t_a is the travel time from station i to station j. t_a is a three-dimensional matrix, as traffic conditions vary at different timestamps, and travel times differ for the two directions between stops. Additionally, travel times between the same two stations can vary at different timestamps. The travel time is expressed as follows:

$$t_a = \begin{Bmatrix} \begin{array}{c|cccc} & t_0 & s_0 & s_1 & \cdots & d_n \\ d_0 & t_{0a_0} & & & & \\ s_0 & & t_{0a_1} & & & \\ \vdots & & & \ddots & & \\ s_n & & & & & t_{0a_n} \end{array} & \begin{array}{c|cccc} & t_0 & s_n & s_{n-1} & \cdots & d_0 \\ d_n & t_{0a_{n+1}} & & & & \\ s_n & & t_{0a_{n+2}} & & & \\ \vdots & & & \ddots & & \\ s_0 & & & & & t_{0a_{2n}} \end{array} \end{Bmatrix} \times T$$

The travel time between stations, for example, from timestamp t_0, the travel time from stop s_n to depot d_n is denoted as t_{0a_n}. And there will be T (the number of all timestamps) matrices of this type.

A detailed description of nodes, arcs and paths is provided. Nodes are divided into stations and depots. Arcs are divided into travel arcs, delivery arcs, transfer arcs, and waiting arcs. Paths are formed by connecting these arcs in the order of stations and timestamps.

(1) Nodes

Depot Node: The set of these nodes is denoted as N_d, indexed by $n_d : (d, t)$. These nodes represent the starting and ending stations of all routes. For example, (d_1, t_0) and (d_2, t_0) represent the starting and ending points of Line 1 at the first timestamp. Each route has a pair of corresponding depots as starting and ending points at each timestamp.

Stop Node: The set of these nodes is denoted as N_s, indexed by $n_s : (s, t)$. These nodes represent all stops except the starting and ending stations of fixed routes. For example, (s_1, t_0) is the stop node for the first stop of Line 1 at the first timestamp. Each stop on each route at each timestamp has a unique identifier. Delivery and pick-up nodes outside the fixed routes start after the numerical identifiers of the stops on all routes. Some stops on fixed routes are transfer stops, meaning that although the station numbers are different on different routes, their physical locations are the same.

(2) Arcs

Traveling Arc: The set of these arcs is denoted as A_t, describing the movement of MAUs between stations on a fixed route, indexed by (i, t, j, t'). i and j are stops only on the fixed route or depots.

$$A_t \in \{(i,t,j,t') \,|i = (d_n, s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}),\ j = \left(s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}, \ldots, d_n\right), t' = t + t_a\}$$

Delivery Arc: The set of these arcs is denoted as A_d, describing the movement of MUs between delivery nodes outside the fixed routes or connecting to stops or depots on the fixed routes, indexed by (i, t, j, t'). The attributes are similar to those of travel arcs, but the connected stations are different.

$$A_d \in \{(i,t,j,t') \,|i = (d_n, s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}),\ j = \left(s_n, s_{n+1}, \ldots, d_{n+1}, s_{n+m}, \ldots, d_n\right), t' = t + t_a\}$$

Transfer Arc: The set of these arcs is denoted as A_r, describing the movement of MAUs between stops on different routes that share the same physical location, indexed by (i, t, j, t'). i and j are predefined stops, and MAUs can move freely between transfer nodes. $t' = t + \Delta$, where Δ is the predetermined transfer time required for MAUs.

$$A_r \in \{(i,t,j,t') \,|i = (s_n, s_{n+m}),\ j = (s_{n+m}, s_n), t' = t + \Delta\}$$

Waiting Arc: The set of these arcs is denoted as A_w, describing the state where MAUs remain stationary, indexed by (i, t, i, t'). i represents predefined stations where MAUs can wait, including

depots, transfer stops, and all delivery stops outside the fixed routes. $t' = t + \Omega$, where Ω is the predetermined waiting time.

$$A_w \in \{(i, t, i, t') \,|i = (d_n, s_n), t' = t + \Omega\}$$

(3) Paths

Each path is composed of a certain number of different types of arcs, and each path has two directions: forward and backward. The starting point of a path should begin at a depot and end at a depot. Additionally, the second set of stations and timestamps of the previous arc must match the first set of stations and timestamps of the subsequent arc. Only when both conditions are satisfied can a feasible path be formed. And since passengers can only be transported on fixed routes, all paths are further divided into passenger paths and freight paths. Passenger paths include all depots and stops on a fixed bus line, while freight paths additionally include all delivery and pick-up stops.

4.2. Demand

This study assumes that daily passenger and freight demand are known in advance. Expressing demand as the quantity on each arc is due to the difficulty in predicting precise transportation needs for specific passengers or freight, while arc-based demand, derived from statistical data, is more reliable and easier to obtain. Additionally, although the demand model does not capture specific passenger or freight boarding and alighting behaviors but represents them in groups (number of passengers or freight arriving at a station within a certain period of time), this approach is more suitable for scenarios with substantial transportation demand within a region.

For passenger demand, since the bus lines' schedules are known, demand is concentrated before the bus is expected to arrive. Therefore, from timestamp t to timestamp t', if there are n passengers with transportation demand from stop s to stop s', it will be represented as $e_i = \{(s, t, s', t') : n\}$. For freight demand, due to its lower time sensitivity, it only needs to be transported within a certain time window. If there are n pieces of freight that need to be sent out from stop s between timestamp (t, t') and arrive at stop s' between timestamp (t'', t'''), it will be represented as $f_i = \{(s, (t, t'), s', (t'', t''')) : n\}$. Due to the greater time flexibility of freight, this will enable more efficient scheduling strategies for MAU across different time periods in a day.

4.3. Problem illustrations

To clearly illustrate the trajectories of MAUs in this study for transporting passengers and freight within a Space-Time network, as shown in Figure 5, a group of MAUs conducted a trip that simultaneously accommodated Fixed-route transit (FRT) and Demand-responsive transit (DRT),

demonstrating the flexibility of MAUs in a multi-route passenger-freight integrated transport system. Additionally, Figure 6 supplements the location and load status of each MAU at every timestamp. For the location status, for example, d_0 indicates that the MAU is stationed at depot d_0 at this timestamp, while s_1, s_2 indicates that the MAU is traveling between stops s_1 and s_2 at this timestamp.

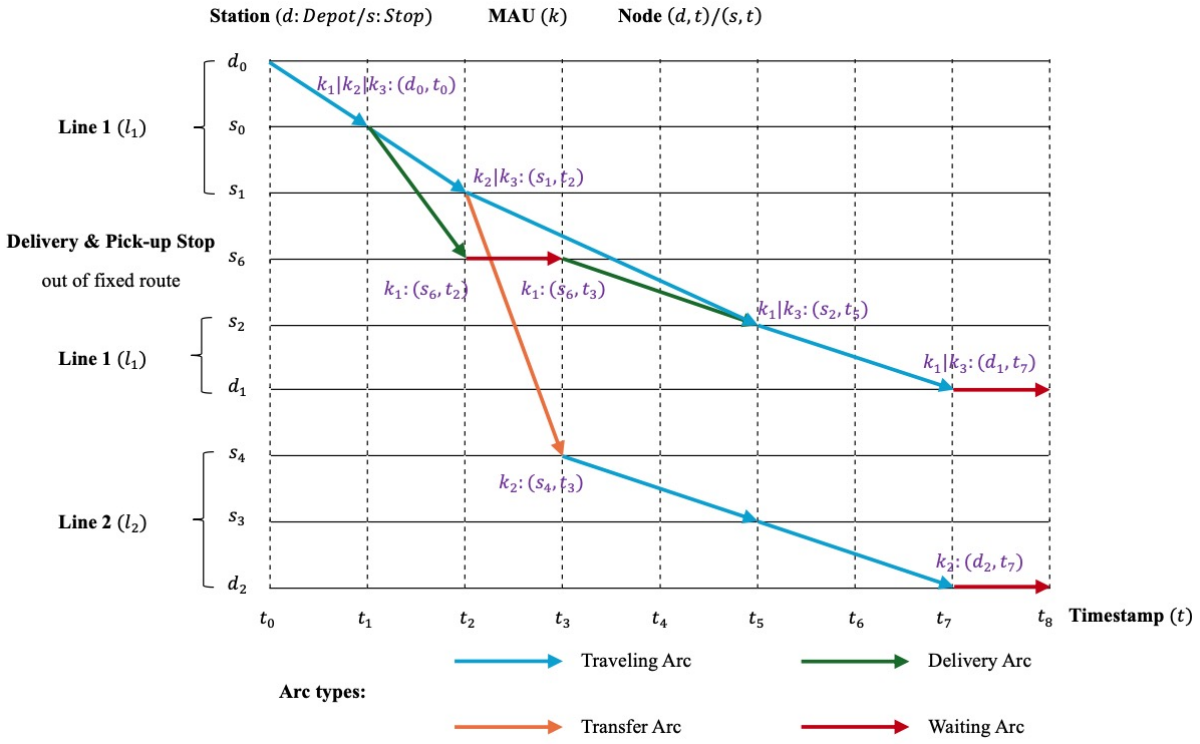

Figure 5 **MAU's transport trajectory on the Space-Time Network**

Timestamps / MAU id	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
MAU k_1	d_0:	s_0:	s_6:	s_6:	s_6, s_2:	s_2:	s_2, d_1:	d_1:	d_1:
MAU k_2	d_0:	s_0:	s_1:	s_4:	s_4, s_3:	s_3:	s_3, d_2:	d_2:	d_2:
MAU k_3	d_0:	s_0:	s_1:	s_1, s_2:	s_1, s_2:	s_2:	s_2, d_1:	d_1:	d_1:

MAU Carriage Status: : Passenger only : Freight only : Passenger & Freight : Empty

Figure 6 **MAU's Position and Loading/Unloading status at each timestamp**

Three assembled MAUs, numbered k_1, k_2, and k_3, form a MAV, and depart simultaneously from the depot d_0 on Line 1, starting at node (d_0, t_0). At the starting point, k_1 carries both passengers and freight, while k_2 and k_3 transport only passengers. After traveling for one unit of timestamp via the travel arc (d_0, t_0, s_0, t_1), they arrive at the next stop. At this point, k_2 and k_3 have sufficient capacity to handle the demand from s_1 to s_2, allowing k_1 to fulfill a demand-responsive transit within the time window. The passengers in k_1's carriage transfer to k_2 or k_3, enabling k_1 to detach and perform a delivery task. At timestamp t_2, k_1 arrives at the delivery and pick-up stop s_6 and unloads all freight in its carriage, while k_2 and k_3 arrive at s_1. Since k_3 has sufficient capacity to meet the demand on (s_1, t_2, s_2, t_5), the idle k_2 can transfer to Line 2 via the transfer arc (s_1, t_2, s_4, t_3) to continue service, transporting passengers from the geographical locations s_1 and s_4 to s_3 via (s_4, t_3, s_3, t_5). The passenger and freight demand on (s_2, t_5, d_1, t_7) exceeds k_3's capacity. After waiting for one timestamp via the waiting arc (s_6, t_2, s_6, t_3), the idle k_1 travels to s_2 and reassembles with k_3 into a single MAU at t_5. Ultimately, k_1 and k_3 arrive at the terminal d_1 on Line 1 at t_7, while k_2 arrives at the terminal on Line 2 at t_7.

In this example, although all three MAUs depart from node (d_0, t_0) on Line 1, they take three different paths to complete the transportation tasks. Through dynamic formation, line switching, and spatiotemporal coordination, the MAUs significantly enhance the operational efficiency and adaptability of passenger and freight co-transportation in modular transit systems.

5. Mathematical formulations

This thesis proposes a mathematical model for the MAU Routing Problem of Multi-functional Autonomous Units (MAUs) in a passenger-freight Space-Time Network, aiming to minimize total transportation costs while satisfying passenger and freight demand constraints. The problem is based on a network structure comprising depots, stops, and timestamps, considering MAU path selection, passenger and freight transport capacity constraints, and restrictions on MAU formations on the same arc. The model determines the path allocation for utilized MAUs, the fulfillment of passenger and freight demands, and the balance of MAUs entering and leaving depots. The following mathematical model is formulated as a Mixed-Integer Quadratically Constrained Programming (MIQCP) problem, integrating the objective function and constraints to ensure efficient optimization of transportation plans in complex networks.

The problem involves Multi-Depot, Space-Time Network, MAUs, and passenger-freight intermodal transportation. Since the Path-based Model pre-generates feasible paths p, it offers greater flexibility and conciseness compared to other models, making it the preferred choice for addressing this problem. Firstly, because the network incorporates both time and space dimensions, the path-based model eliminates the need for additional constraints to ensure continuity in space and time, thereby simplifying optimization decisions. At the same time, path-based decision-making aligns better with the formation flexibility of MAVs. By selecting paths, MAUs avoid real-time computation of all possible arc combinations, reducing complexity. Furthermore, since passenger transport operates only on fixed routes while freight is picked up and delivered at all nodes, capacity allocation becomes more intuitive. This approach allows for a clear definition of whether a path is carrying passengers or freight.

In the following sections, all sets, parameters and decision variables are introduced in detail and the complete mathematical model is presented.

5.1. Notations

All sets establish the foundation for entities and their relationships within the MAU transportation system, creating a structured representation for indexing vehicles K, paths P, demands $E\&F$, and physical infrastructure. Parameter settings quantify the system's operational characteristics and constraints, including demand volumes $\gamma_e\&\lambda_f$, vehicle capacity Q, and more. By adjusting these parameters, different scenarios can be simulated to observe the impact of changes in demand or capacity on decision-making. The core decision variable $y^k_{e,a}$, which the model seeks to optimize, determines the allocation of vehicles on paths and directly influences costs. The auxiliary variables $y^k_{e,a}$ and $z^k_{f,a}$ play a critical role in tracking the passenger/freight loading status of vehicles and the fulfillment of demand quantities on each arc. Table 3 shows the notations and explanations of the sets, parameters, and decision variables used in the model

Table 3 Notation and Description for the Model

Notation	Description
Sets	
D	Set of depots, $D = \{1, 2, ..., \|D\|\}$, indexed by d
S	Set of stops, $S = \{1, 2, ..., \|S\|\}$, indexed by s
I	Set of stations, $I = \{1, 2, ..., \|I\|\}$, indexed by i, j
T	Set of timestamps, $T = \{1, 2, ..., \|T\|\}$, indexed by t, t
N	Set of nodes, $N = \{(1, 1), (2, 2), ..., \|I, T\|\}$, indexed by n
A	Set of arcs, $A = \{(1, 1, 2, 2), (2, 2, 2, 3), ..., \|I, T, J, T'\|\}$, indexed by a
P	Set of paths of MU k, $P = \{(at_1, ..., \|A_t\|, \|A_d\|, \|A_r\|, \|A_w\|), ...\}$, indexed by p
K	Set of MAUs, $K = \{1, 2, ..., \|K\|\}$, indexed by k
E	Set of sectional passenger demand, $E = \{1, 2, ..., \|E\|\}$, indexed by e
F	Set of freight group demand, $F = \{1, 2, ..., \|F\|\}$, indexed by f
Parameters	
c_p	Costs of path p
Q	Maximum capacity of each MAU k
G	Maximum number of MAU formations on the same travel arc or delivery arc
ρ	Passenger and freight capacity occupancy ratio
γ_e	Total volume of sectional passenger demand e
λ_f	Total volume of freight group demand f
$B^-_{d,p}$	Number of MAUs leaving depot d in path p
$B^+_{d,p}$	Number of MAUs entering depot d in path p
θ^p_a	Binary, if arc a is in the path p, the value is 1, otherwise 0
Decision Variables	
x^k_p	Binary, if MAU k passes through the path p, the value is 1, otherwise 0
$y^k_{e,a}$	Integer, number of sectional passenger demand e carried by MAU k on arc a
$z^k_{f,a}$	Integer, number of freight group demand f carried by MAU k on arc a

5.2. Objective and Constraints

$$\min \quad \sum_{k \in K} \sum_{p \in P} c_p x^k_p \tag{1a}$$

$$\text{s.t.} \quad \sum_{p \in P} x^k_p \leq 1 \quad \forall k \in K \tag{1b}$$

$$y^k_{e,a} \leq \gamma_e \sum_{p \in P} \theta^p_a x^k_p \quad \forall k \in K, e \in E, a \in A \tag{1c}$$

$$z_{f,a}^k \leq \lambda_f \sum_{p \in P} \theta_a^p x_p^k \quad \forall k \in K, f \in F, a \in A \tag{1d}$$

$$\sum_{a \in A} \sum_{k \in K} y_{e,a}^k = \gamma_e \quad \forall e \in E \tag{1e}$$

$$\sum_{a \in A} \sum_{k \in K} z_{f,a}^k = \lambda_f \quad \forall f \in F \tag{1f}$$

$$\sum_{p \in P} \sum_{e \in E} \theta_a^p y_{e,a}^k x_p^k + \frac{1}{\rho} \sum_{p \in P} \sum_{f \in F} \theta_a^p z_{f,a}^k x_p^k \leq Q \quad \forall k \in K, a \in A \tag{1g}$$

$$\sum_{k \in K} \sum_{p \in P} \theta_a^p x_p^k \leq G \quad \forall a \in A \tag{1h}$$

$$x_p^k \in \{0,1\} \quad \forall k \in K, p \in P \tag{1i}$$

$$y_{e,a}^k \in [0, \gamma_e] \quad \forall k \in K, e \in E, a \in A \tag{1j}$$

$$z_{f,a}^k \in [0, \lambda_f] \quad \forall k \in K, f \in F, a \in A \tag{1k}$$

Objective Function: The objective (1a) is to minimize the cost of all paths that MAU k passes through. Since the cost is directly linked to the length of travel time, this means that the solution which this study is looking for is to assign each vehicle the lowest total path travel time.

Path Assignment Constraint: Constraint (1b) ensures that each MU k select at most one path p.

Demand Capacity Constraints: Constraint (1c) ensures that for each MU k, the volume of sectional passenger demand e transported does not exceed the total amount of that demand γ_e, and that transport is allowed only when the path p selected by the MAU k supports the sectional passenger demand. Constraint (1d) ensures that for each MU k, the volume of freight group demand f transported does not exceed the total amount of that demand λ_f, and that transport is allowed only when the path p selected by the MAU k supports the freight group demand.

Demand Satisfaction Constraints: Constraint (1e) ensures that all sectional passenger demands γ_d are met. Constraint (1f) ensures that all freight group demands λ_f are met.

Vehicle Capacity Constraint: Constraint (1g) ensures each MAU k on the path p, its total load (passengers and freight) shall not exceed the capacity Q. At the same time, ρ is added to flexibly adjust the different capacity occupancy rules between passengers and freight.

Vehicle Grouping Constraint: Constraint (1h) ensures that the number of MAU formation on each arc does not exceed G. During peak hours, the MAU can be controlled to prioritize passenger demand rather than freight demand.

The optimization model comprehensively considers multiple key constraints to ensure the accuracy of path allocation, providing a feasible verification framework for the subsequent design of the ALNS algorithm. Path allocation, demand capacity, demand satisfaction, flow conservation, and vehicle capacity and grouping constraints collectively form the feasible region boundaries of the solution.

6. Solution methods

When selecting a computational method suitable for this MAU Routing Problem, it is considered that the model needs to simultaneously optimize the routing of each MAU, platoon formation, the integration of passenger and freight, and fleet size. This introduces a large number of binary decision variables (x_p^k) for assigning and integer variables ($y_{e,a}^k$ and $z_{f,a}^k$) for demand allocation, along with nonlinear quadratic constraints such as capacity limits, which increase the complexity of the solution process. For small-scale instances, exact solvers like GUROBI can find optimal solutions within an acceptable time frame. However, for medium or large-scale instances, exact methods relying on branch-and-bound algorithms face exponential growth in solution time as the number of variables increases for NP-hard problems, potentially leading to memory shortages. In practical operations, since transportation planning requires near-real-time rapid decision-making, the computational speed of exact solvers is insufficient, whereas heuristic methods can provide near-optimal solutions with minimal gaps in a shorter time.

Compared to other heuristics, the Adaptive Large Neighborhood Search (ALNS) algorithm can balance diversification and intensification, avoiding the risk of getting trapped in local optima while also maintaining convergence speed. The adaptive mechanism of ALNS allows for dynamic weight adjustments based on operator performance and enables the destruction and repair of large solution structures, enhancing search efficiency and solution diversity. Furthermore, the ALNS algorithm can be adjusted to account for modular formation characteristics, making it particularly suitable for scenarios where passenger and freight are integrated within the same MAU.

Based on these observations, this section focuses on designing a hybrid Adaptive Large Neighborhood Search (ALNS) algorithm at the arc and path levels to find high-quality solutions within acceptable computational time. Two initial solution generation methods are compared. First, a small number of feasible paths are generated based on demand, and then GUROBI or heuristic methods are used to obtain initial solutions. Building on this, ALNS is employed to explore feasible solutions that meet demand with lower costs. The overall process of the algorithm is shown in the Figure 7:

For this MAV Routing Problem, the solution of the ALNS algorithm must pass the following four feasibility checks:

1. All passenger and freight demands are satisfied without exceeding the capacity of each MAU;
2. The carriage of each MAU can be in states with only passengers, only freight, or both passengers and freight simultaneously;
3. After completing the transportation demand task on an arc, the MAU's capacity must be released, meaning that a single MAU should be able to fulfill multiple demands along a single path;
4. The number of MAU groups on each arc cannot exceed the limit.

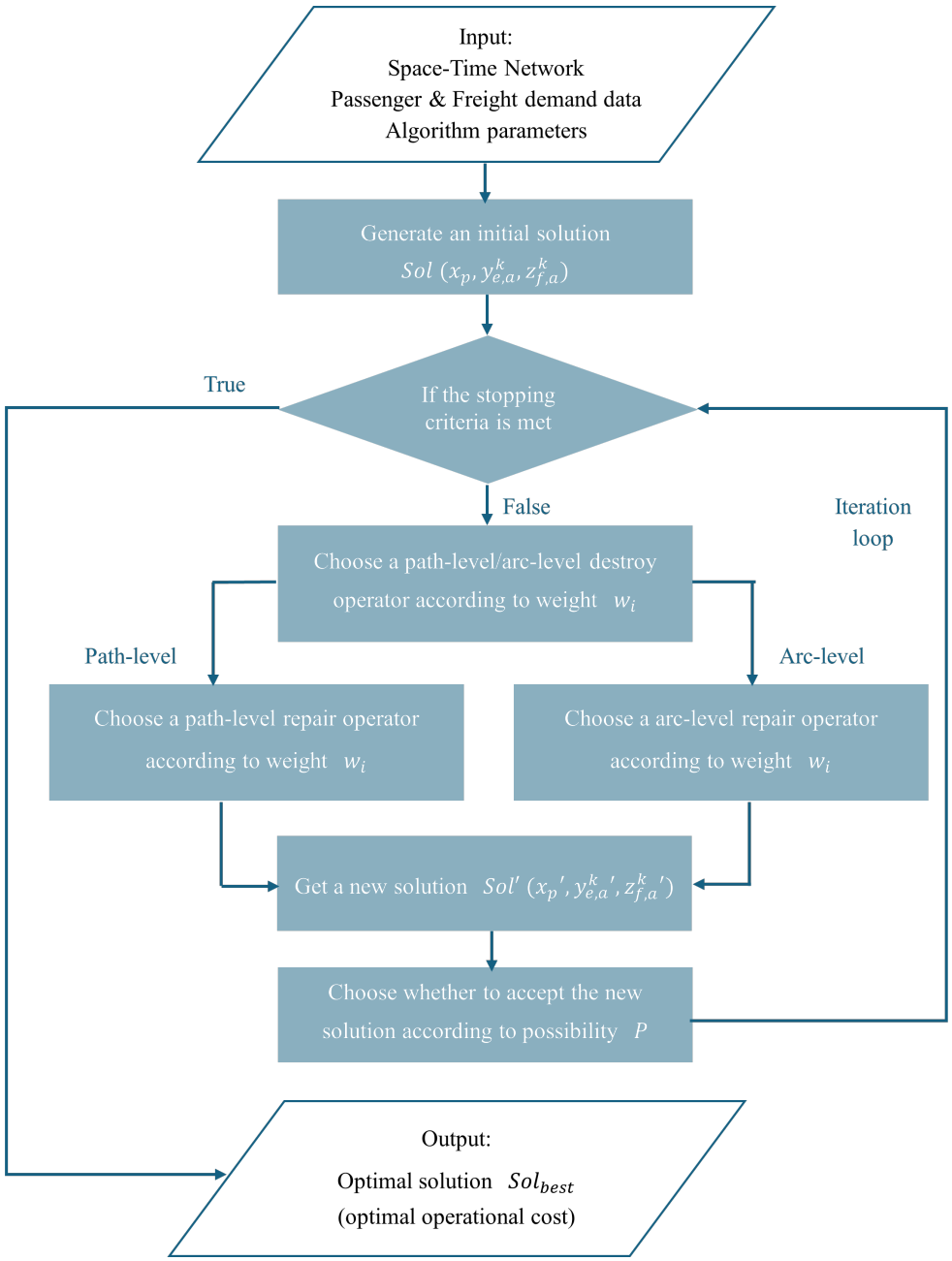

Figure 7 **Overall flowchart of algorithm**

6.1. Adaptive Large Neighborhood Search algorithm

Shaw (1997) first proposed a Local Search algorithm that employs greedy local search and utilizes a larger neighborhood to avoid local minima. Ropke and Pisinger (2006) extended this approach, improving it into the Adaptive Large Neighborhood Search (ALNS) algorithm. Traditional local search techniques typically explore only a limited subset of solutions, making only small modifications to the current solution and easily getting trapped in local optima. ALNS, however, can remove

a significant portion of the solution and reconstruct it in a single iteration, while also allowing the use of multiple destruction and repair methods within the same search process, providing greater flexibility in finding optimal solutions.

ALNS consists of three main components: an initial solution, destruction/repair operations, and an acceptance criterion. The ALNS algorithm starts with an initial solution, which serves as the starting point for the iterative search process. In each iteration, the algorithm performs a series of operations aimed at improving the current solution. These operations include destruction (removing a portion of the current solution using a specified removal operator) and repair (reinserting the removed portion using an insertion operator). The solution obtained after each iteration is called the incumbent solution and is evaluated through the acceptance criterion to determine whether it is accepted. Throughout the iterative process, the algorithm continuously updates the current solution based on the results of the destruction and repair operations, enabling ALNS to explore different regions of the solution space to find better solutions.

The score updating procedure tracks the performance of each heuristic algorithm, guiding the search process toward the most promising regions of the solution space. In each iteration, removal and insertion heuristic algorithms are applied to the current solution, and the scores of these operators are continuously updated. If a heuristic algorithm finds a new global best solution or discovers an unvisited solution that is accepted by the acceptance criterion, its score increases. However, if it performs poorly, failing to improve the solution or explore new regions, its score may decrease or remain unchanged. A higher score indicates that the heuristic algorithm is more successful in finding good solutions. This dynamic scoring mechanism ensures that the algorithm prioritizes recently high-performing heuristics, thereby improving search efficiency and the quality of solutions.

The new solution generated by modifying the current solution through the application of removal and insertion heuristic algorithms is referred to as the proposed solution. Whether to accept or reject this solution is determined probabilistically, inspired by the Simulated Annealing (SA) algorithm, based on the difference in the objective function values between the current solution Sol and the proposed solution Sol', as well as the current temperature parameter T. Figure 8 illustrates the framework of the ALNS algorithm in this study.

6.2. Acceptance criterion

The ALNS algorithm uses Simulated Annealing (SA) as the acceptance rule. The main idea of SA in ALNS is not only to accept improved solutions, but also to provide opportunities for accepting worse solutions. The SA process starts with a high initial temperature, allowing the acceptance of worse solutions to explore the solution space. As the process continues, the temperature gradually

Input: initial solution $Sol\ (x_p^k, y_{e,a}^k, z_{f,a}^k)$
While $i < \max Iteration$ **do**:
1 Select a destroy operator according to the probabilistic score
2 Select an insertion operator according to the probabilistic score
3 Generate a new solution Sol'
4 If Sol' is accepted then
$Sol = Sol'$
End
5 If $f(Sol) < f(Sol')$ then
$Sol_{best} = Sol$
End
6 Update the scores of the operators
7 Update the temperature & acceptance probability
8 $i = i + 1$
Output: Best solution Sol_{best}
End

Figure 8 Framework of the ALNS algorithm

decreases, tending towards accepting improved solutions. According to Metropolis guidelines, if the objective function value of the new solution $f(Sol)'$ is better than the current solution $f(Sol)$, the new solution is accepted. Otherwise, a random number $rand$ between 0 and 1 is generated, and the new solution is accepted with probability $P = e^{-\frac{f(Sol')-f(Sol)}{T}} > rand$. This probabilistic mechanism allows the algorithm to jump out of local optima in the early stage and gradually converge to high-quality solutions in the later stage. After each iteration, the temperature T is updated using the expression $T \cdot \psi$, where ψ is the cooling rate. In this study, the initial temperature T_0 is set to 100, and the cooling rate ψ is 0.995. The principles used by SA in the ALNS algorithm are shown in Figure 8.

6.3. Generation of initial solution

In order to shorten the time for generating the initial solution, both methods use a small number of feasible paths for decision making, which are called demand paths. All feasible paths are called valid paths, denoted as P_{valid}. Demand paths are a subset of valid paths, extracted from valid paths based on arcs with transportation demands, denoted as P_{demand}. This can effectively reduce the number of decision variables that need to be considered in the initialization method.

(1) GUROBI solver method

Although the GUROBI solver performs poorly with large-scale data, it can quickly and accurately obtain optimal decisions for small-scale data, providing a better starting point for ALNS compared to random generation or simple heuristics. This allows ALNS to converge to a satisfactory solution

Input: initial solution $x_p^k, y_{e,a}^k, z_{f,a}^k$, initial temperature T, cooling rate ψ
While $i < \max Iteration$ **do**:
1 Generate a new solution Sol' after destroy and insertion processes
2 If $P = e^{-\frac{f(Sol')-f(Sol)}{T}} > rand(0,1)$ then
$Sol = Sol'$
End
3 If $f(Sol) < f(Sol')$ then
$Sol_{best} = Sol$
End
4 Update T
5 $i = i + 1$
Output: Best solution Sol_{best}
End

Figure 9 Simulated Annealing in ALNS algorithm

with fewer iterations, significantly reducing overall runtime. Additionally, solutions derived from the mathematical model always pass feasibility checks, avoiding the difficulty of finding feasible initial solutions under complex constraints in a short time. This hybrid strategy of exact solver and heuristic algorithms fully leverages their complementary strengths: GUROBI's mathematical optimization capability quickly identifies a high-quality starting point, while ALNS's neighborhood search ability further explores and improves the solution space.

(2) Greedy heuristic method

This method maximizes the coverage capability of MAUs through iterative allocation to satisfy unassigned demand on paths. Priority is given to selecting paths with high coverage-to-cost ratios based on demand, where the cost can be balanced using this formula to obtain paths that satisfy the remaining transportation demand requirements:

$$s_p = \frac{1}{\sqrt{c_p}} \left(\sum_{e \in E} u_e \sum_{a \in P} \alpha_e^a + \frac{1}{\rho} \sum_{f \in F} u_f \sum_{a \in P} \beta_f^a \right) \tag{2}$$

where $u_e = \gamma_e - \sum_{a \in A} \sum_{k \in K} y_{e,a}^k$ represents the unassigned passenger demand, α_e^a is a binary variable indicating whether this arc can transport this passenger demand, and $\sum_{e \in E} u_e \sum_{a \in P} \alpha_e^a$ represents the total unassigned passenger demand that the path can cover. Similarly, $u_f = \lambda_f - \sum_{a \in A} \sum_{k \in K} z_{f,a}^k$ represents the unassigned freight demand, β_f^a is a binary variable indicating whether this arc can transport this freight demand, and $\frac{1}{\rho} \sum_{f \in F} u_f \sum_{a \in P} \beta_f^a$ represents the total unassigned freight demand that path p can cover.

A three-stage process is adopted: First, vehicles are allocated to paths to satisfy passenger demand. Under the constraint of adhering to capacity limits, MAUs with available capacity can satisfy freight demand, as first satisfy:

$$y_{e,a}^{k} \leq \min(u_e, Q - \sum_{e \in E} y_{e,a}^{k}) \quad \forall k \in K, a \in A \tag{3}$$

then satisfy:

$$z_{f,a}^{k} \leq \min\left(u_f, Q - \sum_{e \in E} y_{e,a}^{k} - \frac{1}{\rho} \sum_{f \in F} z_{f,a}^{k}\right) \quad \forall k \in K, a \in A \tag{4}$$

THen, all MAUs assigned to the same arc are checked for grouping situations. All demands must be satisfied under mandatory constraints, as $u_e = 0$ and $u_f = 0$. However, exceeding the maximum fleet size for a single arc may occur. This happens because this constraint are not subject to high-priority requirements. In subsequent ALNS algorithms, this can be effectively addressed through disruption and insertion processes. This iterative startup method can find an initial solution in a very short time and can greatly accelerate the solution speed.

6.4. Destroy heuristics

This study categorizes destroy operators into two levels: path and arc. Path-level destruction removes entire paths, which can release large solution spaces and significantly reconstruct MAU path allocation, while Arc-level destruction preserves useful portions of paths and more precisely removes arcs that have no demand but occupy costs. Each type of operator has its own advantages and disadvantages, working together to maintain the depth of local optimization while providing the breadth of global search. This section explains the rationale and principles behind using these ALNS removal heuristic methods.

1. Dynamic destroy rate

The destruction rate is a parameter that controls the proportion of vehicle paths removed from the solution in each iteration, denoted by ξ and constrained within $[\min \xi, \max \xi]$. This ensures that each destruction operation has a certain degree of disturbance, avoiding the algorithm being too conservative while preventing excessive destruction that would lead to complete randomization of the solution. When no better solution is found after a certain number of iterations, the destruction rate is increased using $\xi' = \min(\max \xi, \xi + 0.05)$ to encourage the algorithm to explore new solution spaces. When a better solution is found, the counter is reset to 0, and the destruction rate is decreased using $\xi' = \max(\min \xi, \xi - 0.02)$ to make the algorithm more inclined to perform fine optimization within the neighborhood of the current solution. The dynamic change process is shown in Figure 10. The target number of vehicle allocation paths to be removed for each operator is:

$$Sol_{target} = \max(1, \lfloor Sol_{active} \times \xi \rfloor) \tag{5}$$

where Sol_{active} is the set of vehicles that have already been allocated paths in the current solution. However, the actual number of removals is constrained to avoid completely destroying the solution by:

$$Sol_{remove} = \min(Sol_{target}, \lfloor Sol_{active} - 1 \rfloor) \tag{6}$$

Input: initial destroy rate ξ , destroy rate range $[\min \xi, \max \xi]$

While $i < \max Iteration$ **do**:

1 When solution Sol has no improvement:
  no improvement count =+ 1
  When no improvement count $\geq$ n
    $\xi' = \min(\max \xi, \xi + 0.05)$
  no improvement count = = 0

2 When solution Sol has an improvement:
    $\xi' = \max(\min \xi, \xi - 0.02)$
  no improvement count = = 0

End

Figure 10 **Dynamic destroy rate process**

2. Path-level destroy operator

(1) Random Destroy

The core principle of Random Destroy is to disrupt the structure of the current solution by randomly selecting and removing a certain number of vehicle routes. Although it is the most basic destruction strategy, it is highly important. The primary reason for using this operator is its ability to provide essential diversity to the algorithm. When trapped in a local optimal solution, it can unbiasedly explore various corners of the solution space, effectively preventing premature convergence. Additionally, randomness helps the algorithm escape its current search trajectory, offering fresh reconstruction opportunities for the subsequent repair phase. This random perturbation mechanism is key to maintaining the algorithm's long-term search vitality and often unexpectedly discovers new regions containing high-quality solutions. The specific steps are shown in Figure 11.

(2) Worst Cost Destroy

The design principle of Worst Cost Destroy is based on a key optimization intuition: the highest-cost routes often represent the least efficient resource allocations in the current solution, making their removal most likely to yield significant cost improvements. This operator is distinctly goal-oriented, targeting routes with suboptimal costs and providing the algorithm with a clear direction

Input: Current Solution Sol
  Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$
1 **Select** $|Sol_{remove}|$ paths *randomly*
2 **Remove**
Output: Partially Destroyed Solution Sol'
  End

Figure 11 **Random Destroy Process**

for improvement. Since a few high-cost routes typically contribute disproportionately to the total cost, removing them can create the greatest potential for improvement in the subsequent repair phase. Moreover, the presence of high-cost routes often suggests that the current route selections may be suboptimal, indicating the potential for alternative solutions with more efficient resource utilization. The specific steps are shown in Figure 12.

Input: Current Solution Sol
  Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$
1 **Sort** cost of each path of x_p^k
2 **Select** top $|Sol_{remove}|$ *highest-cost* paths
3 **Remove**
Output: Partially Destroyed Solution Sol'
  End

Figure 12 **Worst Cost Destroy Process**

(3) Demand-based Destroy

The design principle of Demand-based Destroy is rooted in the concept of optimizing transportation efficiency, which involves reconfiguring vehicle routes with low demand service efficiency, as identified by the key index 'efficiency = number of transportation demand / path cost'. During off-peak periods, there may be vehicle routes with relatively low passenger and cargo loads or low service demand density. Even if these routes have modest costs, low-efficiency vehicles often indicate resource waste, suggesting suboptimal route planning that results in insufficient vehicle loads. By removing these inefficient vehicles, the algorithm creates opportunities to reorganize and optimize demand allocation, forcing the repair phase to seek solutions that better utilize vehicle capacity and serve higher-value demands more effectively. The specific steps are shown in Figure 13.

3. Arc-level destroy operator

(1) Arc Removal Destroy

Input: Current Solution Sol
Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$
1 **Calculate** efficiency for each path of x_p^k $(Efficiency(p_i) = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}])$
2 **Select** top $|Sol_{remove}|$ *lowest-efficiency* paths
3 **Remove**
Output: Partially Destroyed Solution Sol'
End

Figure 13 **Demand-based Destroy Process**

The core of Arc Removal Destroy is to remove a certain number of non-demand arcs from an allocated path, and then select the segment containing the most demand from the disconnected segments as the new path. First, the paths with the most non-demand arcs are selected. For a path, the target number of non-demand arcs to be removed is controlled within the range $A_{target} = \text{random}\,(1, \xi \times A_{active})$, where A_{active} refers to the total number of arcs contained in this path, and the actual number of removals satisfies $A_{remove} = \min(A_{target}, A_{non-demand})$, where $A_{non-demand}$ refers to the number of all non-demand arcs on this path. After completing the removal steps, there will be multiple segments containing demand remaining, and only the segment containing the most demand is retained to avoid the situation where arcs with demand have only one unique connectable non-demand arc, providing more exploration space in the repair phase. The specific steps are shown in Figure 14.

(2) Arc Search Destroy

Arc Search Destroy is a destruction operator based on transportation efficiency analysis of each arc. First, similar to Demand-based Destroy, the least efficient paths are found, but the entire path to which the MAU is assigned will not be removed. Then the transportation efficiency of each arc on the path to be destroyed is calculated according to the formula $Efficiency(a_i) = \frac{1}{c_{(a_i)}}[\gamma_{(a_i)} + \frac{1}{\rho}\lambda_{(a_i)}]$, where $\gamma_{(a_i)}$ is the number of passengers transported by arc a_i, $\lambda_{(a_i)}$ is the freight quantity, and $c_{(a_i)}$ is the cost required to pass through this arc. After sorting each arc segment, the ξ least efficient ones will be removed. Since non-demand arcs have a numerator of 0 in the efficiency calculation, most of them will be preferentially removed. If more than half of the arcs serve no demand, random selection will be performed. Through destroying empty arcs or low-value arcs with excessively high costs, this provides a better foundation for subsequent repair operators. The specific steps are shown in Figure 15.

6.5. Insertion heuristics

To repair the removed vehicle paths or arc segments, ALNS requires insertion heuristic methods to regain new complete solutions. This study employs three path-level repair operators: Random

Input: Current Solution Sol

Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$

Sort non-demand arcs contained in each path of x_p^k

Select top $|Sol_{remove}|$ *most non-demand arcs* paths

For each selected path:

  Select all non-demand arcs:

    Set $|A_{remove}| = min(A_{target}, A_{non-demand})$,
    where $A_{target} = random(1, \xi \times A_{active})$

    Select $|A_{remove}|$ *non-demand* arcs

    Remove

  Identify segments of the path containing demand:

    Select the segment with *most demand arcs*

      Append the segment as a new path into Sol'

Output: Partially Destroyed Solution Sol'

End

Figure 14 **Arc Removal Destroy Process**

Input: Current Solution Sol

Select number of x_p^k to be destroyed: $|Sol_{remove}| = |Sol| \times \xi$

Calculate efficiency for each path ($Efficiency(p_i) = \frac{1}{C_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$)

Select top $|Sol_{remove}|$ *lowest-efficiency* paths

For each selected path:

  Calculate efficiency for each arc ($Efficiency(a_i) = \frac{1}{C_{(a_i)}}[\gamma_{(a_i)} + \frac{1}{\rho}\lambda_{(a_i)}]$)

  Sort arcs by Efficiency

  Select $|A_{remove}|$ *lowest-efficiency* arcs

  Remove

  Identify segments of the path remaining:

    Append the segments as a new path into Sol'

Output: Partially Destroyed Solution Sol'

End

Figure 15 **Arc Search Destroy Process**

Repair, Greedy Repair, and Utility Maximization Repair, along with three arc-level repair operators: Arc Insertion Repair, Chain Repair, and Hybrid Repair. The destroy operators at path and arc levels are matched with corresponding repair operators at the same level to ensure repair quality. This section explains the reasons and principles behind using these ALNS insertion heuristic methods.

1. Candidate paths

To improve algorithm efficiency, after certain demands are removed by the destroy operator, the algorithm does not blindly search through all possible paths. Instead, it prioritizes paths that are highly related to the removed demands, known as candidate paths. During the repair process, candidate paths serve two main roles. First, the search space is significantly reduced, excluding many meaningless paths and thereby substantially accelerating the algorithm's runtime. Second, this study establishes a path relevance scoring mechanism, where paths capable of serving multiple removed demands receive higher scores, enhancing the quality of the repair.

2. Path-level repair operator

(1) Random Repair

The working principle of Random Repair in this study involves selecting the feasible candidate paths which can cover the deleted demands and performing random selection under the premise of feasibility. The primary reason for using this operator is its ability to effectively counteract the search stagnation that may occur in the ALNS algorithm during prolonged iterations. When the solution structures remain similar over an extended period, random repair can break this regularity, maintaining the algorithm's creativity. The specific steps are shown in Figure 16.

Input: Partially Destroyed Solution Sol'
  Generate candidate paths which can cover destroyed demand
  Iterate while u_e & $u_f \neq 0$:
1 **Select** the path *randomly*
2 **Insert**
  Evaluate feasibility
Output: Repaired Solution Sol
  End

Figure 16 **Random Repair Process**

(2) Greedy Repair

Greedy Repair evaluates the value of each candidate path using the efficiency index 'number of transportation demand / path cost', prioritizing the allocation of vehicles to paths that can serve the most demands at the lowest cost. The algorithm calculates the demand-to-cost ratio for each candidate path to obtain its service value score, selecting the path with the highest input-output efficiency for priority allocation. While the greedy strategy may not guarantee a global optimum, it offers high computational efficiency, enabling the rapid repair of an incomplete solution post-destruction into a complete feasible solution. The specific steps are shown in Figure 17.

(3) Utility Maximization Repair

Input: Partially Destroyed Solution Sol'
  Generate candidate paths which can cover destroyed demand
  Iterate while u_e & $u_f \neq 0$:
1 **Calculate** efficiency for each candidate path ($Efficiency(p_i) = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$)
2 **Select** the *highest-efficiency* path
3 **Insert**
  Evaluate feasibility
Output: Repaired Solution Sol
  End

Figure 17 Greedy Repair Process

Utility Maximization Repair selects paths with higher opportunity costs, where high opportunity cost means that the path achieves a good balance between predicted passenger capacity and cost. For each candidate path in the solution pool that can repair the removed demand, the utility value is first calculated through $Utility_{(p_i)} = \frac{1}{c_{(p_i)}}[(\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}) \times (1 + B_{capacity})]$, where $\gamma_{(p_i)}$ is the maximum number of passengers that can be transported on the path, $\lambda_{(p_i)}$ is the maximum amount of freight that can be transported, $c_{(p_i)}$ is the cost of this path, and $B_{capacity}$ is the capacity utilization rate reward used to expand the advantage of paths with higher average capacity utilization rates. Then the regret value is calculated as $Regret_{(p_i)} = (Utility_{max} - Utility_{(p_i)}) - 0.1 \times \max(0, c_{max} - c_{(p_i)})$, where $Utility_{max}$ is the highest utility value among all paths, and c_{max} is the highest cost among all paths. The regret value is a quantification of decision risk, with lower regret values representing better input-output ratios for this path investment. Finally, a comprehensive score is used to find the optimal trade-off between each path's utility and risk control, with the scoring formula being $Utility\ score_{(p_i)} = (0.7 \times \frac{Utility_{(p_i)}}{Utility_{max}}) + (-0.3 \times \frac{Regret_{(p_i)}}{Regret_{max}})$. Paths with higher scores will be preferentially selected and allocated demand in a progressive manner, seeking to allocate the highest-scoring path among the remaining paths that can satisfy unallocated demand. The specific steps are shown in Figure 18.

3. Arc-level repair operator

(1) Arc Insertion Repair

Arc Insertion Repair can find the optimal insertion scheme for removed demand, enabling paths to maintain feasibility while minimizing insertion costs. Each removed arc with demand is assigned a fixed time window, and value ranking is performed through passenger quantity $\gamma_{(a_i)}$ and freight quantity $\lambda_{(a_i)}$. Arcs with more passenger demand and higher value will be preferentially selected, followed by attempting all possible insertion positions in the currently destroyed paths through enumeration. After successfully inserting one position, the remaining demand will continue to

Input: Partially Destroyed Solution Sol'

Generate candidate paths which can cover destroyed demand

Iterate while u_e & $u_f \neq 0$:

1 **Calculate** utility for each candidate path:

$$(Utility_{(p_i)} = \frac{1}{c_{(p_i)}}[(\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}) \times (1 + B_{capacity})])$$

2 **Calculate** regret value for each candidate path:

$$(Regret_{(p_i)} = \left(Utility_{max} - Utility_{(p_i)}\right) - 0.1 \times \max(0, c_{max} - c_{(p_i)}))$$

3 **Calculate** utility score for each candidate path:

$$(Utility\ score_{(p_i)} = (0.7 \times \frac{Utility_{(p_i)}}{Utility_{max}}) + (-0.3 \times \frac{Regret_{(p_i)}}{Regret_{max}}))$$

4 **Select** the *highest-scoring* path

5 **Insert**

Evaluate feasibility

Output: Repaired Solution Sol

End

Figure 18 **Utility Maximization Repair**

attempt insertion into this path. When no suitable arc can be inserted, the lowest-cost non-demand arc is found to repair the path to complete feasibility, calculating the cost required to use this path. After evaluating the insertion cost of each position, the most efficient scheme is selected, as the path with the highest score in this formula: $Efficiency_{(p_i)} = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$. Since this operator exhaustively searches all possible insertion combinations, it can ensure finding locally optimal solutions. The specific steps are shown in Figure 19.

(2) Chain Repair

Chain Repair outputs optimized and reorganized complete paths by inputting all broken arc segments and unsatisfied demand. Its objective is $\min \sum_{p_i \in P}(\sum_{fragments(p_i)} c_{fragments(p_i)} + \sum_{connections(p_i)} c_{connections(p_i)})$, where $connections(p_i)$ includes arcs with demand as well as non-demand arcs used for connections. This operator selects the path set with the minimum total cost from all feasible combinations, with the capability of combining multiple paths, optimizing from an overall reconstruction perspective. The specific steps are shown in Figure 20.

(3) Hybrid Repair

When an originally short path has a certain number of arcs removed, the length of remaining arc segments may not meet the preservation conditions, leading to the entire path being removed, while some paths are not completely destroyed, resulting in mixed destruction. When mixed destruction occurs, the algorithm can intelligently identify and assign the Hybrid Repair operator for restoration. For paths with existing segments, Arc Insertion Repair is used, and if there is demand that

Input: Partially Destroyed Solution Sol'

Identify arcs which have destroyed demand & **Assign** each demand arc a time window

Iterate while u_e & $u_f \neq 0$:

1. **Rank** demand arcs by number of passengers $\gamma_{(a_i)}$ and freight $\lambda_{(a_i)}$
2. **Select** *currently highest-ranked* arc:
3. **Enumerate** all possible insertion position by the time window on a partially destroyed path & **Insert**
4. **Continue** searching remaining demand arcs by the time window & **Insert** if the arc fit the time window
5. **Find** lowest-cost non-demand arcs to connect & **Evaluate** feasibility
6. **Collect** all possible repaired paths
7. **Calculate** efficiency for each repaired path ($Efficiency(p_i) = \frac{1}{c_{(p_i)}}[\gamma_{(p_i)} + \frac{1}{\rho}\lambda_{(p_i)}]$)
8. **Select** the *highest-efficiency* path
9. **Insert**

Output: Repaired Solution Sol

End

Figure 19 **Arc Insertion Repair Process**

Input: Partially Destroyed Solution Sol'

Identify arcs which have destroyed demand & all paths with broken arc segments

1. **Set** the target:

$$min \sum_{p_i \in P'} \Big(\sum_{fragments(p_i)} c_{fragments(p_i)} + \sum_{connections(p_i)} c_{connections(p_i)} \Big)$$

2. **Generate** all feasible path sets P'
2. **Select** *minimum total cost* path set P'

Output: Repaired Solution Sol

End

Figure 20 **Chain Repair Process**

cannot be allocated, path-level Greedy Repair is employed for allocation. This operator can first handle high-certainty demand, then process remaining demand with minimal waste. The specific steps are shown in Figure 21.

6.6. Operator Selection Mechanism

The Destroy and Insertion operators adopt a roulette wheel selection strategy, which maintains the weight of each operator to dynamically adjust the selection probability. Specifically, the selection probability of operator i is calculated by the formula $P_i = \frac{w_i}{\sum_{j=1}^{n} w_j}$, where w_i is the current weight of operator i, and $\sum_{j=1}^{n} w_j$ is the sum of all operator weights of the same type. All operators

Input: Partially Destroyed Solution Sol'
Identify all paths if broken arc segments exist:
1 Path with segments exist:
Apply *Arc Insertion Repair*
2 Path with no segment exist:
Apply *Greedy Repair*
Output: Repaired Solution Sol
End

Figure 21 Hybrid Repair Process

have an initial weight of 1.0, ensuring that each operator has an equal chance of being selected at the beginning of the algorithm. When an operator combination produces an improvement, the corresponding operator weights are updated according to $w_i^{t+1} = w_i^t \times (1+\alpha)$; when no improvement is produced, the weights are updated according to $w_i^{t+1} = w_i^t \times (1-0.5\alpha)$, where α is the weight update factor. This selection mechanism enables superior operator combinations to obtain higher selection probabilities, thereby improving overall search efficiency.

7. Numerical experiments

To verify the performance of the proposed approaches, this section first tests some small-scale data to examine the gap and effectiveness between the algorithms, and then conducts numerical experiments on a real case study. The proposed algorithm is coded in Python on a Windows 11 personal computer with 13th Gen Intel(R) Core(TM) i7-13700H and 32G RAM. GUROBI 12.1.1 is used to solve the MAV Routing Problem model.

7.1. Impact of Space-Time Network scale on different solution methods

The scale of the model is primarily determined by the number of stations and timestamps. An increase in the number of timestamps and stations both leads to an increase in the number of nodes in the Space-Time Network, which in turn causes a rise in the number of paths and, consequently, more demand. This section divides the experiments into two parts: fixed timestamps and fixed station numbers, to compare the solution quality and computational speed of different methods. Additionally, the number of fixed stations will be further divided into two aspects: changes in the number of lines and changes in the number of pickup and delivery stops.

The calculation of costs involves 1 timestamp representing 1 minute. MAU's per-minute travel cost depending on its energy consumption, travel speed, and local electricity price. The average energy consumption of MAU is 22 kWh/100km (Solar Impulse Foundation 2025), and it is assumed to operate at an average speed of 30km/h in urban settings (including driving and stopping). The average price for public charging in the Netherlands is €0.43/kWh (EV Connect 2025). The per-minute electricity cost for MAU can be calculated using the following formula:

$$\text{Cost (€/min)} = \frac{\text{Energy consumption (kWh/100km)} \times \text{Speed (km/h)}}{100} \times \text{Electricity cost (€/kWh)} \tag{7}$$

The result is a cost of €0.0473/min. Regarding other parameters, based on the NEXT Company database (NExT Modular Vehicles 2024), A MAU carriage has a capacity Q of 15 passengers, and this study assumes that 2 pieces of freight occupy the capacity of 1 passenger, as ρ is 2. Therefore, the same carriage can accommodate 30 pieces of freight. Additionally, to prevent MAV formation on the road due to the MAU's composition, the number of coupled units G is limited to 3. Furthermore, due to the limited number of timestamps, the travel time is set to be relatively compact, ranging between 1 and 3 minutes.

To investigate the impact of increasing the number of lines on the performance of different algorithms, the timestamp is fixed at 20. As shown in Table 4, the first column represents the parameters adjusted in the experiment, denoted as $T - L - PD$, where T is the number of timestamps, L is the number of fixed bus lines, and this experiment uses 2, 3, and 4 lines, with each line fixed at 8 stations, including 2 depots and 6 intermediate stops. Among them, stops s_2 and s_7 are

geographically identical and connected by a transfer arc. The delivery and pick-up stops outside the lines are fixed at 4, denoted as PD. Figure 22 illustrates this space network using 2 lines as an example. The second column in Table 4 represents the number of paths to be traversed in the current experiment, the fourth column represents the optimal solution obtained by the algorithm, which indicates the total travel time of all MAUs in this network. The fifth column represents the total operational cost calculated based on the travel time. The sixth and seventh columns represent the time taken to obtain the initial solution and the time spent by the algorithm, respectively. The eighth column represents the total computation time to complete the experiment.

Table 4 Comparison of algorithms for different number of lines

Instance (T-L-PD)	Number of paths traversed	Solution method	Objective		Computational time (sec)		
			Travel time (min)	Cost (euro)	Initial solution	ALNS algorithm	Total
20 - 2 - 4	40280	GUROBI	–	–	–	–	>21600.0
		GUROBI+ALNS	154	7.28	204.5	23.9	228.4
		Greedy heuristic+ALNS	162	7.66	0.3	28.3	28.6
20 - 3 - 4	67327	GUROBI	–	–	–	–	–
		GUROBI+ALNS	198	9.37	471.3	33.2	504.5
		Greedy heuristic+ALNS	214	10.12	0.4	52.6	53.0
20 - 4 - 4	84301	GUROBI	–	–	–	–	–
		GUROBI+ALNS	290	13.72	1181.7	184.4	1366.1
		Greedy heuristic+ALNS	328	15.51	0.9	201.5	202.4

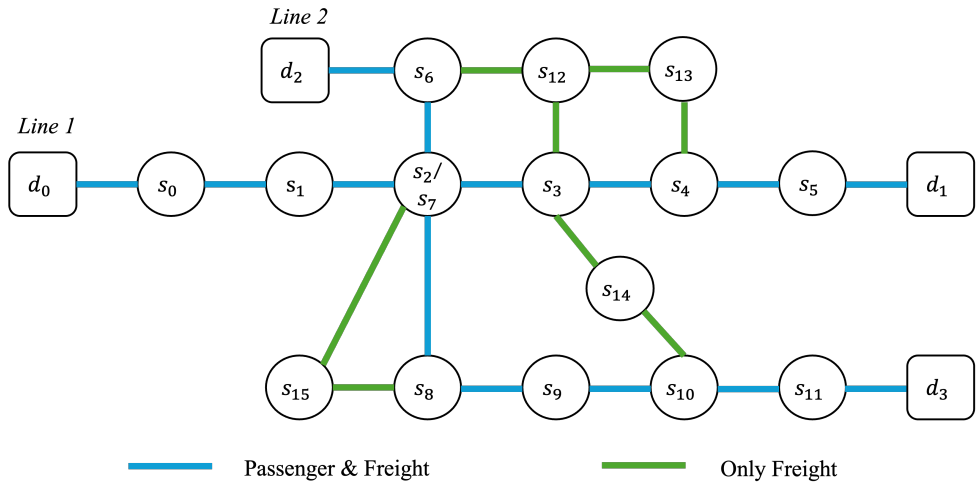

Figure 22 Spatial network for experiments with fixed timestamp numbers (20-2-4)

Due to the large number of stations and the dispersed nature of demand, the GUROBI solver struggles to find any solution within 6 hours for a network with 2 lines, 15 passenger demands,

and 5 freight demands. In terms of initial solution computation time, the Greedy heuristic method is significantly faster than the GUROBI method, which solves a small-scale path problem with demand. However, the initial solution quality of the GUROBI method is better, leading to shorter subsequent computation times for ALNS. The computation time of the GUROBI method increases exponentially with the number and dispersion of demands, resulting in a growing gap in total computation time compared to the Greedy heuristic + ALNS algorithm as the network size expands. In addition, in the 20-2-4 experiment, the gap between the optimal solutions obtained by the two methods was 5.2%, while in the 20-4-4 experiment, the gap reached 10.3%. The solution quality obtained by the GUROBI+ALNS algorithm is consistently better than that of the Greedy heuristic+ALNS algorithm.

Next, the experiment fixes the number of lines at 2, still using 20 timestamps, with the number of pick-up and delivery stops set to 2, 4, and 6 cases, while the passenger demand remains unchanged, and the freight demand increases proportionally with the number of pick-up and delivery stops. For example, the connection scenario with 4 pick-up and delivery stops uses the same spatial network as the 20-2-4 configuration, as shown in Figure 22. As shown in Table 5, since the stops outside these fixed lines can serve to connect different lines, compared to the stops on the fixed lines, each additional pick-up and delivery stop causes the number of paths to be traversed to grow exponentially.

Table 5 Comparison of algorithms for different number of pick-up and delivery stops

Instance (T-L-PD)	Number of paths traversed	Solution method	Objective		Computational time (sec)		
			Travel time (min)	Cost (euro)	Initial solution	ALNS algorithm	Total
20 - 2 - 2	6876	GUROBI	134(78.4)	6.34	–	–	1714.6
		GUROBI+ALNS	136	6.43	19.2	8.5	31.7
		Greedy heuristic+ALNS	140	6.62	0.2	9.1	9.3
20 - 2 - 4	40280	GUROBI	–	–	–	–	>21600.0
		GUROBI+ALNS	154	7.28	204.5	23.9	228.4
		Greedy heuristic+ALNS	162	7.66	0.3	28.3	28.6
20 - 2 - 6	92478	GUROBI	–	–	–	–	–
		GUROBI+ALNS	208	9.84	2099.1	87.1	2186.2
		Greedy heuristic+ALNS	237	11.2	2.3	113.1	115.4

In the smallest-scale experiment, the GUROBI solver obtained the optimal solution, with the GUROBI+ALNS method achieving an optimal solution with a gap of 1.5%, and the Greedy heuristic+ALNS method at 4.5%. Combined with the previous experiment that varied the number of lines, it can be observed that GUROBI is highly sensitive to the spatial network's expansion in

terms of computational speed. When using GUROBI to generate initial solutions, it is evident that adding two more pick-up and delivery stops increases the computation time by approximately tenfold. In contrast, the heuristic methods are less affected. To further validate the gap with the exact solutions obtained by the GUROBI solver, the number of stations was reduced to decrease the dispersion of demand across the spatial network.

To further verify the gap between the exact solution obtained by the GUROBI solver, the number of timestamps was reduced to decrease the spatial dispersion of demand. A fixed number of stations is used, and the algorithm's quality was tested by progressively increasing the number of timestamps. As shown in Figure 23, the experiment utilized two fixed lines. Additionally, there were two delivery and pick-up stops, namely s_{12} and s_{13}, where s_{12} can connect to s_3 and s_6, and s_{13} can connect to s_2, s_7, and s_8. As shown in Table 6, this experiment used 12 groups of tests, with timestamps ranging from 10 to 120, increasing by intervals of 10.

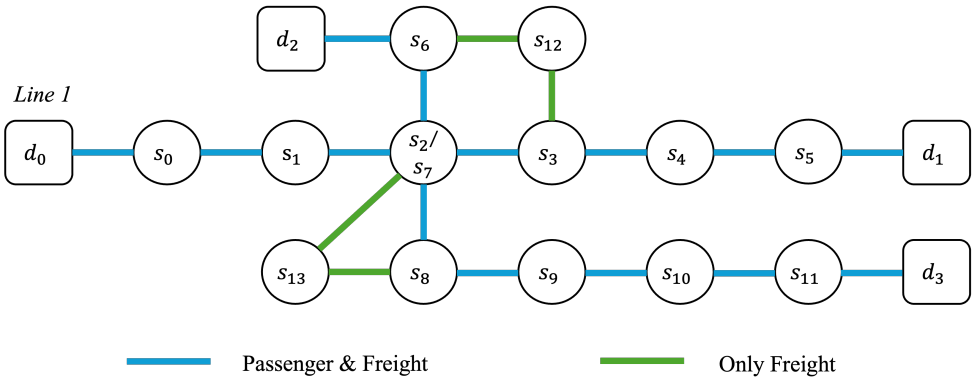

Figure 23 **Spatial network for fixed station number experiments**

Taking the experiment with 10-2-2 as an example, Table 7 shows the paths assigned to each MAU in the Space-Time network and the demand on each corresponding arc. From the results, it can be observed that some demands are split, or the remaining capacity of the carriages is utilized for freight transportation. Regarding the composition of MAVs, for instance, k_1 and k_2 form an MAV in some segments of the path, while k_3 and k_4, for example, travel together throughout the entire journey. Additionally, for example, k_9 does not perform any transportation tasks and is an MAU traveling empty to balance depot flow.

In small-scale tests, the GUROBI solver is able to provide exact solutions. In the Objective column, the exact integer solution obtained by the GUROBI solver is accompanied by the relaxed solution in parentheses. In two sets of experiments, the gaps between these two solutions were 17.9% and 32.4%, respectively, fully demonstrating the complexity of the model. Through comparison, the solution quality of both the GUROBI+ALNS algorithm and the Greedy heuristic method+ALNS

Table 6 Comparison of algorithms for different number of timestamps

Instance (T-L-PD)	Number of paths traversed	Solution method	Objective		Computational time (sec)		
			Travel time (min)	Cost (euro)	Initial solution	ALNS algorithm	Total
10 - 2 - 2	153	GUROBI	69(56.7)	3.26	–	–	2.2
		GUROBI+ALNS	69	3.26	1.1	2.8	3.9
		Greedy heuristic+ALNS	71	3.45	0.2	3.1	3.2
20 - 2 - 2	26905	GUROBI	102(69.0)	4.82	–	–	1440
		GUROBI+ALNS	103	4.92	23.6	20.5	44.1
		Greedy heuristic+ALNS	107	5.20	0.3	27.3	27.6
30 - 2 - 2	250244	GUROBI	–	–	–	–	>21600.0
		GUROBI+ALNS	155	7.33	48.7	216.6	265.3
		Greedy heuristic+ALNS	155	7.33	0.4	233.6	234.0
40 - 2 - 2	429590	GUROBI	–	–	–	–	–
		GUROBI+ALNS	159	7.52	82.2	297.5	379.5
		Greedy heuristic+ALNS	163	7.71	0.6	420.1	420.7
50 - 2 - 2	654375	GUROBI	–	–	–	–	–
		GUROBI+ALNS	180	8.51	538.6	394.6	933.2
		Greedy heuristic+ALNS	187	8.85	1.1	602.8	603.9
60 - 2 - 2	840069	GUROBI	–	–	–	–	–
		GUROBI+ALNS	212	10.03	630.2	662.6	1292.8
		Greedy heuristic+ALNS	215	10.17	1.3	788.1	789.4
70 - 2 - 2	1049975	GUROBI	–	–	–	–	–
		GUROBI+ALNS	244	11.54	1085.0	691.5	1776.5
		Greedy heuristic+ALNS	258	12.20	1.7	1403.9	1405.6
80 - 2 - 2	1247367	GUROBI	–	–	–	–	–
		GUROBI+ALNS	292	13.81	1864.5	704.8	2569.3
		Greedy heuristic+ALNS	303	14.33	2.3	2040.6	2042.9
90 - 2 - 2	1463988	GUROBI	–	–	–	–	–
		GUROBI+ALNS	376	17.78	2068.9	1187.6	3256.5
		Greedy heuristic+ALNS	391	18.49	3.1	2543.8	2546.9
100 - 2 - 2	1677193	GUROBI	–	–	–	–	–
		GUROBI+ALNS	402	19.01	3249.7	1781.5	5031.2
		Greedy heuristic+ALNS	435	20.58	4.0	3367.8	3371.8
110 - 2 - 2	1859714	GUROBI	–	–	–	–	–
		GUROBI+ALNS	424	20.06	4953.2	2414.9	7368.1
		Greedy heuristic+ALNS	458	21.66	4.8	3750.7	3755.5
120 - 2 - 2	2073843	GUROBI	–	–	–	–	–
		GUROBI+ALNS	445	21.05	10772.5	2987.1	13759.6
		Greedy heuristic+ALNS	460	21.79	7.6	6876.9	6884.5

algorithm fell within an acceptable range, as shown in Figure 24. In the 10-2-2 experiment, the gap between the GUROBI+ALNS algorithm's solution and GUROBI's solution was 0.0%, while the gap for the Greedy heuristic method+ALNS algorithm was 2.9%. In the 20-2-2 experiment, the gap for the GUROBI+ALNS algorithm was 0.9%, while the gap for the Greedy heuristic method+ALNS algorithm was 4.7%. Although the GUROBI+ALNS algorithm requires more time

Table 7 Experimental results example (10-2-2)

MAU id and Demand	Arc: start station (start timestamp) → end station (end timestamp)						
k_0	$d_0(0) \rightarrow s_0(2)$	$s_0(2) \rightarrow s_1(4)$	$s_1(4) \rightarrow s_2(6)$	$s_2(6) \rightarrow d_1(8)$			
Passenger demand	$e_0:10$	-	-	$e_3:15$	-	-	-
Freight demand	$f_0:10$	$f_1:30$	-	-	-	-	-
k_1	$d_0(0) \rightarrow s_0(2)$	$s_0(2) \rightarrow s_1(4)$	$s_1(4) \rightarrow s_2(6)$	$s_2(6) \rightarrow d_1(8)$			
Passenger demand	$e_0:10$	$e_1:15$	$e_2:15$	$e_3:15$	-	-	-
Freight demand	-	-	-	-	-	-	-
k_2	$d_2(1) \rightarrow s_3(2)$	$s_3(2) \rightarrow s_4(4)$	$s_4(4) \rightarrow s_5(6)$	$s_5(6) \rightarrow d_3(8)$			
Passenger demand	$e_7:10$	$e_8:5$	$e_{10}:5$	$e_11:5$	-	-	-
Freight demand	-	$f_2:20$	$f_5:10$	$f_6:10$	-	-	-
k_3	$d_0(2) \rightarrow s_0(3)$	$s_0(3) \rightarrow s_6(4)$	$s_6(4) \rightarrow s_0(5)$	$s_0(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_4(7)$	$s_4(7) \rightarrow s_3(8)$	$s_3(8) \rightarrow d_2(9)$
Passenger demand	$e_{12}:15$	-	-	$e_9:10$	$e_{14}:10$	-	-
Freight demand	-	$f_7:10$	$f_8:10$	$f_9:10$	-	-	$f_{10}:20$
k_4	$d_0(2) \rightarrow s_0(3)$	$s_0(3) \rightarrow s_6(4)$	$s_6(4) \rightarrow s_0(5)$	$s_0(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_4(7)$	$s_4(7) \rightarrow s_3(8)$	$s_3(8) \rightarrow d_2(9)$
Passenger demand	$e_{12}:10$	$e_{13}:10$	-	$e_9:15$	$e_{14}:5$	$e_{15}:5$	$e_{16}:5$
Freight demand	-	-	-	-	-	-	$f_{10}:10$
k_5	$d_2(3) \rightarrow s_3(4)$	$s_3(4) \rightarrow s_4(5)$	$s_4(5) \rightarrow s_5(6)$	$s_5(6) \rightarrow d_3(8)$			
Passenger demand	-	-	-	$e_8:15$	-	-	-
Freight demand	-	-	$f_4:25$	-	-	-	-
k_6	$d_1(3) \rightarrow s_2(5)$	$s_2(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$			
Passenger demand	$e_4:5$	-	$e_5:5$	-	-	-	-
Freight demand	-	$f_{13}:10$	-	-	-	-	-
k_7	$d_3(3) \rightarrow s_5(5)$	$s_5(5) \rightarrow s_4(6)$	$s_4(6) \rightarrow s_1(7)$	$s_1(7) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$		
Passenger demand	-	-	-	-	-	-	-
Freight demand	-	$f_{12}:15$	$f_{11}:30$	$f_{14}:10$	$f_3:5$	-	-
k_8	$d_3(3) \rightarrow s_5(5)$	$s_5(5) \rightarrow s_4(6)$	$s_4(6) \rightarrow s_1(7)$	$s_1(7) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$		
Passenger demand	$e_{19}:5$	$e_{17}:5$	-	$e_{18}:10$	$e_{20}:5$	-	-
Freight demand	-	$f_{12}:20$	$f_{11}:10$	-	-	-	-
k_9	$d_1(3) \rightarrow s_2(5)$	$s_2(5) \rightarrow s_1(6)$	$s_1(6) \rightarrow s_0(8)$	$s_0(8) \rightarrow d_0(9)$			
Passenger demand	$e_4:15$	-	-	$e_6:10$	-	-	-
Freight demand	-	-	-	$f_3:10$	-	-	-

to solve, its solution quality is superior, closely approaching the exact solution. However, starting from the 30-2-2 experiment, due to the excessive number of paths and demands to be searched, the GUROBI solver failed to produce any results within 6 hours. In contrast, the Greedy heuristic method+ALNS algorithm was able to find a result in 234 seconds. While this method significantly improves computational speed, it sacrifices some solution quality.

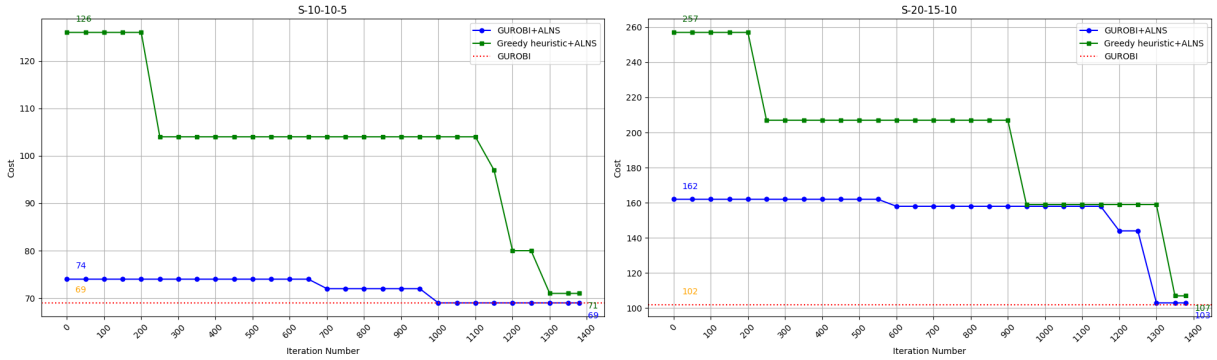

Figure 24 Comparison of the quality of the optimal solution of the algorithms

As the number of timestamps and demands increases, as shown in Figure 25, the computational time gap between the GUROBI+ALNS algorithm and the Greedy heuristic method+ALNS algorithm becomes increasingly significant. In the 70-2-2 experiment, the total time difference between the two methods was 370.9 seconds, while in the 100-2-2 experiment, the time difference expanded to 1659.4 seconds. In the largest-scale experiment, 120-2-2, the time difference between the two methods reached 6875.1 seconds, with the computational time of the GUROBI+ALNS algorithm nearly double that of the Greedy heuristic method+ALNS algorithm. It can be predicted that when the number of stations is fixed, an increase in the number of timestamps will lead to a progressively larger computational time gap between the two methods.

By combining the results of experiments with a fixed number of timestamps and stations, the following observations can be made:

1. Compared with the exact optimal solutions obtained by the GUROBI solver in small-scale experiments, the solutions from the GUROBI+ALNS algorithm are very close to the exact solutions, while the solutions from the Greedy heuristic method+ALNS algorithm exhibit a slightly larger gap but remain within an acceptable range.
2. When the number of timestamps is fixed, increasing the number of stops in fixed lines, pick-up and delivery stops, and demands leads to a progressively more noticeable superiority in the solution quality of the GUROBI+ALNS algorithm compared to the Greedy heuristic method+ALNS algorithm. However, when generating initial solutions, the computational time of the GUROBI method increases exponentially with the number of stations, especially when the number of pick-up and delivery stops increases. In contrast, the computational speed advantage of the Greedy heuristic method+ALNS algorithm will become increasingly evident.
3. When the number of stations is fixed, increasing the number of timestamps and demands does not lead to a significant gap in solution quality between the GUROBI+ALNS algorithm and the Greedy heuristic method+ALNS algorithm. The overall computational time for both

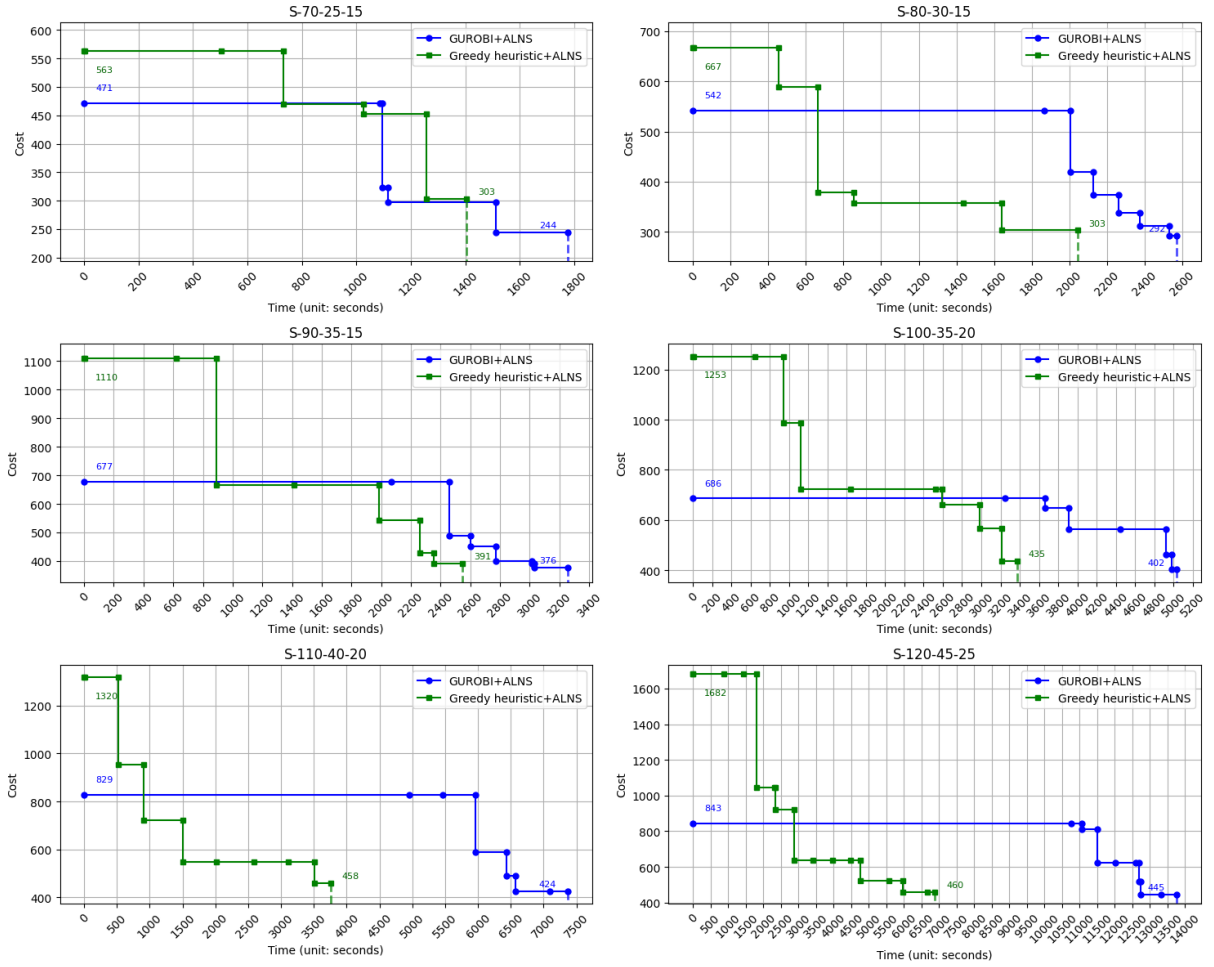

Figure 25 Comparison of algorithms computational time

methods increases steadily, but the Greedy heuristic method+ALNS algorithm remains faster, with the gap widening as the number of timestamps increases.

4. In the Space-Time Network of the MAU Routing Problem, both methods' computational speed of generating initial solutions is influenced by the number of paths to traverse and the number of demands. When expanding through the spatial dimension, the computational time growth rate for both the Greedy heuristic initialization method and the GUROBI initialization method is significantly faster than when expanding through the temporal dimension. This is because, under the influence of the time window of freight demands, a greater number of stations, especially pick-up and delivery stops, leads to a faster increase in path diversity compared to a greater number of timestamps, resulting in a more significant increase in demand paths. The GUROBI initialization method requires more time, which limits the scale of experiments using this method to some extent. However, since the Greedy heuristic initialization method

quickly covers demands based on path scores, the increase in computational time is much less noticeable.

5. The computational time of the ALNS algorithm is insensitive to either dimension (spatial or temporal) and is mostly affected by the quality of the initial solution. Higher-quality initial solutions accelerate the algorithm's convergence process and avoid repeated repair processes due to failed feasibility checks, resulting in shorter computational times. Consequently, the ALNS algorithm using the GUROBI initialization method typically converges faster than when using the Greedy heuristic initialization method. In summary, the ALNS algorithm tailored for the MAU Routing Problem can converge within a limited number of iterations in all instances.

7.2. Sensitivity analysis of passenger and freight capacity occupancy ratio

The passenger and freight capacity occupancy ratio ρ represents the size of the freight, where a larger ρ indicates that the freight occupies less space within the carriage. This section uses a 60-2-2 network to perform a sensitivity analysis on ρ. The cost variations are examined for ρ values of 2, 3, 4, 5, and 6, under the condition that passenger demand and freight demand remain constant and Q is set to 15. As shown in Table 8, the required number of MAUs and the cost results obtained from two initialization methods are recorded.

Table 8 **Result of sensitivity analysis of passenger and freight capacity occupancy ratio**

ρ	Solution method	Number of MAU used	Travel time (min)	Cost (euro)
2	GUROBI+ALNS	19	212	10.03
	Greedy heuristic+ALNS	19	215	10.17
3	GUROBI+ALNS	18	203	9.60
	Greedy heuristic+ALNS	19	207	9.79
4	GUROBI+ALNS	15	178	8.42
	Greedy heuristic+ALNS	15	182	8.61
5	GUROBI+ALNS	14	174	8.23
	Greedy heuristic+ALNS	15	181	8.56
6	GUROBI+ALNS	14	174	8.23
	Greedy heuristic+ALNS	15	180	8.51

Figure 26 illustrates the trend of cost changes under different passenger and freight capacity occupancy ratios. As ρ increases, more freight can be accommodated within the MAU space. Freight that previously required multiple MAUs for transport can now be handled by fewer MAUs, leading to a gradual reduction in the number of MAUs used and, consequently, a decrease in the additional travel costs associated with MAU usage. When $\rho = 4$, a significant reduction is observed, with the number of MAUs used decreasing by 4 compared to when $\rho = 2$, resulting in a cost reduction of

16.1%. This is primarily because, in this experiment, freight demand frequently consists of around 45 to 50 units, while demands exceeding 50 units are rare. Consequently, when $\rho = 5$ or $\rho = 6$, the optimization results show no further cost reduction. The changes in MAU quantity and operational costs caused by ρ primarily depend on the freight demand between pairs of stations and secondarily on the passenger demand within the time window. If the passenger demand on the arcs between two stations within the freight transport time window is consistently high, additional MAUs are still required to transport freight even when ρ is large, making it challenging to achieve significant reductions in optimized operational costs.

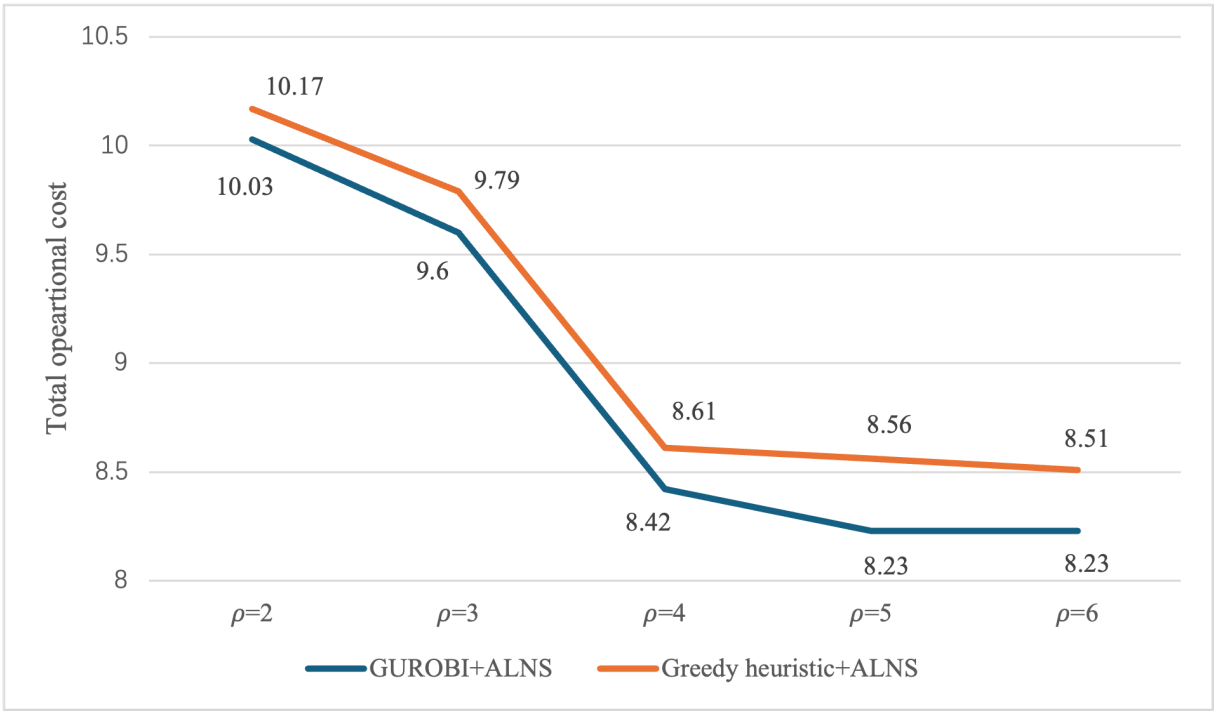

Figure 26 **Operating costs at different passenger and freight capacity occupancy ratios**

7.3. Sensitivity analysis of operators

Among all customized operators, the path-level Utility Maximization Repair operator and the arc-level Chain Repair operator are logically complex, involving multiple layers of functions. The Utility Maximization Repair method requires multiple traversals to evaluate candidate paths, iteratively calculating path utility, capacity utilization, and regret values. Conversely, the Chain Repair operator attempts to combine all remaining path segments into complete paths, involving repetitive path connection checks and demand allocation optimization. These two operators play unique roles in exploring complex solution spaces and optimizing path combinations, but their intricate processes can increase the computational time of the entire ALNS algorithm. To assess the contribution of

these two insertion operators to the overall solution quality and to balance with computational speed, the following experiments were designed.

The experiments were conducted using a spatial layout same as the fixed number of stations, and three groups of experiments with timestamp counts of 50, 60, and 70 were set up for mutual comparison, focusing on the computational time and solution quality changes of the ALNS algorithm. The nine operators other than the Utility Maximization Repair operator and the Chain Repair operator were collectively referred to as basic operators. Table 9 and Table 10 respectively present the computational time and optimal solutions for the GUROBI + ALNS method and the Greedy heuristic + ALNS method under different operator combination conditions.

Table 9 Sensitivity analysis of operators for GUROBI + ALNS method

Operator selection	50-2-2		60-2-2		70-2-2	
	Computational time (sec)	Objective (Travel time)	Computational time (sec)	Objective (Travel time)	Computational time (sec)	Objective (Travel time)
ALNS with basic operators	217.8	225	434.4	265	481.2	305
ALNS with basic operators + Utility Maximization Repair operator	271.9	198	495.6	248	501.9	287
ALNS with basic operators + Chain Repair operator	293.2	182	531.3	220	539.6	249
ALNS with full operators	394.6	180	662.6	212	691.5	244

In terms of computational time, removing these two operators allows the algorithm to complete in nearly half the original time. In the same cases, the Chain Repair operator, which is more complex for global optimization compared to the Utility Maximization Repair operator, often requires more computational time, but the solution quality is very close to that obtained using all operators. In the 60-2-2 experiment with the Greedy heuristic + ALNS method, the solution quality even surpassed that of using all operators, with the largest gap across all experiments being 3.6%. In contrast, although the Utility Maximization Repair operator requires less computational time, its solution quality is poorer, with an average gap of around 9.5% compared to the solution using full operators. Additionally, in several experiments, it exhibited lower weight usage, but still provided some improvement over solution using only basic operators.

The experimental results indicate that omitting the Chain Repair operator significantly reduces solution quality. Although using the complete set of operators generally yields better near-optimal

Table 10 **Sensitivity analysis of operators for Greedy heuristic + ALNS method**

Operator selection	50-2-2		60-2-2		70-2-2	
	Computational time (sec)	Objective (Travel time)	Computational time (sec)	Objective (Travel time)	Computational time (sec)	Objective (Travel time)
ALNS with basic operators	332.5	243	434.4	279	772.1	335
ALNS with basic operators + Utility Maximization Repair operator	415.4	222	543.6	237	965.3	285
ALNS with basic operators + Chain Repair operator	448.8	188	616.3	213	1142.0	261
ALNS with full operators	602.8	187	788.1	215	1403.9	258

solutions in nearly all cases, removing these two operators can be considered when decisions need to be made in a shorter time. In large-scale problems, retaining the Chain Repair operator is more beneficial for maintaining solution quality. As the problem scale increases, the combination of basic operators and the Chain Repair operator saves more time per iteration, making it worthwhile to trade a small portion of solution quality for a 20% reduction in computational time.

7.4. Case study

This section validates the advantages of MAVs compared to traditional buses and delivery vans in the real road network and actual demand scenario of Changning District, Shanghai, China. The spatial network of this study is illustrated in Figure 27. Two regional bus lines were selected, where Line 72 consists of 8 stations $(d_0, s_0, s_1, s_2, s_3, s_4, s_5, d_1)$, and Line 54 consists of 7 stations $(d_2, s_6, s_7, s_8, s_9, s_{10}, d_3)$. Additionally, there are 4 pick-up and delivery stops outside the fixed lines $(s_{11}, s_{12}, s_{13}, s_{14})$. s_2 and s_7 are the intersection points of the two lines, connected via a transfer arc. Among the pick-up and delivery stops outside the fixed lines, s_{11} and s_{12} are interconnected, with s_{11} connected to s_9 and s_{12} connected to s_3. Similarly, s_{13} and s_{14} are interconnected, with s_{13} connected to s_0 and s_1, and s_{14} connected to s_8.

The study selects the morning period from 8:00 to 9:30 as the temporal network, comprising 90 timestamps. The peak period is from 8:00 to 9:00, while the off-peak period is from 9:00 to 9:30. For the same distance, the travel time during the peak period ranges from 4 to 7 minutes, whereas during the off-peak period, it reduces to 2 to 5 minutes. For the fixed lines, the bus timetable operates at a frequency of every 15 minutes during the peak period (8:00, 8:15, 8:30, 8:45) and every 20 minutes during the off-peak period (9:00). Idle MAVs can depart at any time to assist in fulfilling the remaining passenger demand and freight demand. Passenger demand data were

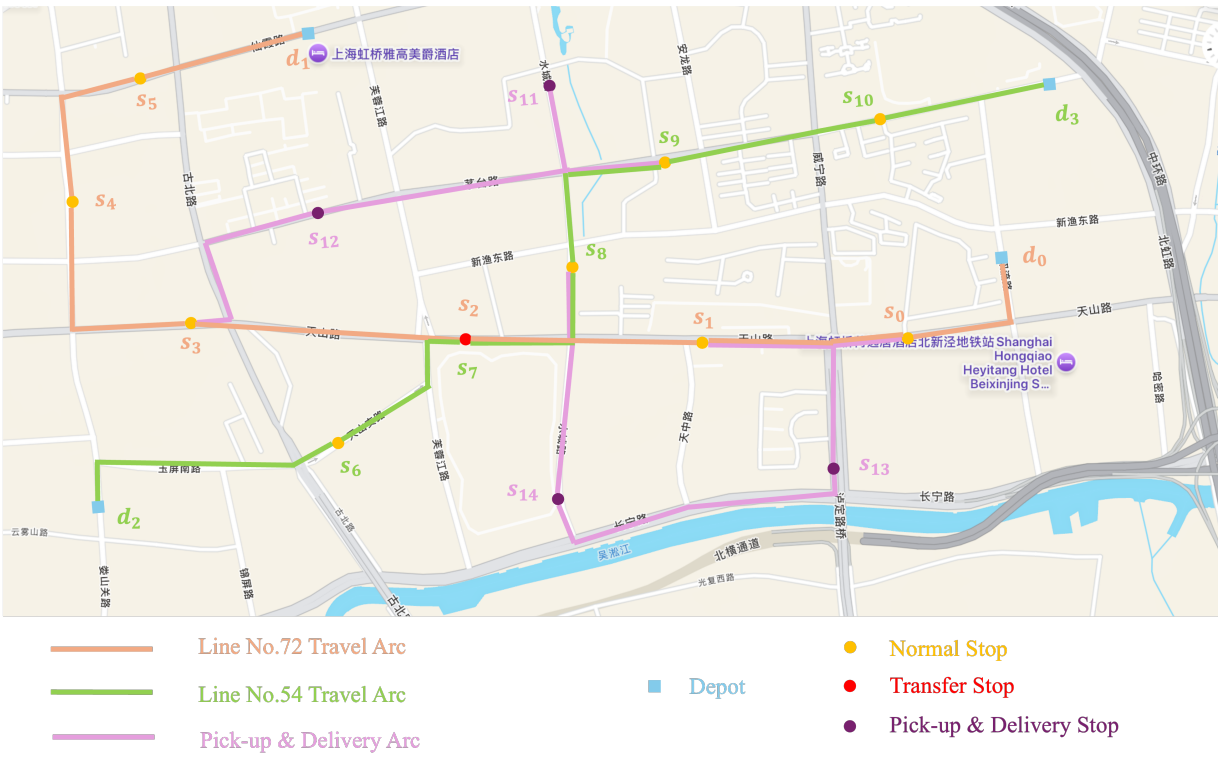

Figure 27 **Spatial network of the case study MAV route in Shanghai region**

obtained from statistics provided by the Shanghai Municipal Transportation Commission. Freight demand data were estimated based on the regional parcel throughput provided by Cainiao with the time window for freight demand spanning all 90 timestamps.

This case study continues to use the parameter information from Section 1, where $Q = 15$, $\rho = 2$, and $G = 3$. As shown in Table 11, due to the wide range of the demand distribution, the GUROBI solver was unable to find any solution within 12 hours. The GUROBI initialization method, even with a reduced number of paths, still failed to produce any results within 8 hours. Therefore, subsequent analyses are based on the solutions obtained from the Greedy heuristic + ALNS algorithm.

Table 11 **Case Result**

Solution method	Objective travel time (min)	Initial solution time (sec)	Algorithm time (sec)	Total computational time (sec)
GUROBI	–	–	–	>43200
GUROBI+ALNS	–	>28800	–	–
Greedy heuristic+ALNS	1054	28.1	8058.2	8086.3

A total of 30 MAUs are deployed in this operation. The detailed path assignments for each MAU are presented in Appendix A. As illustrated in Figure 28, freight transportation is seldom selected during peak hours due to the overwhelming passenger demand. During off-peak periods, each MAU has more remaining capacity, which can be better utilized by freight. During off-peak periods, traditional bus system, in order to balance passenger waiting times, find it difficult to significantly reduce the frequency of departures, resulting in a large amount of redundant capacity in the vehicles. To better demonstrate the flexible utilization of space by MAU, Figure 29, taking k_{25} as an example, illustrates the entire process of fulfilling both partial passenger demand and freight demand.

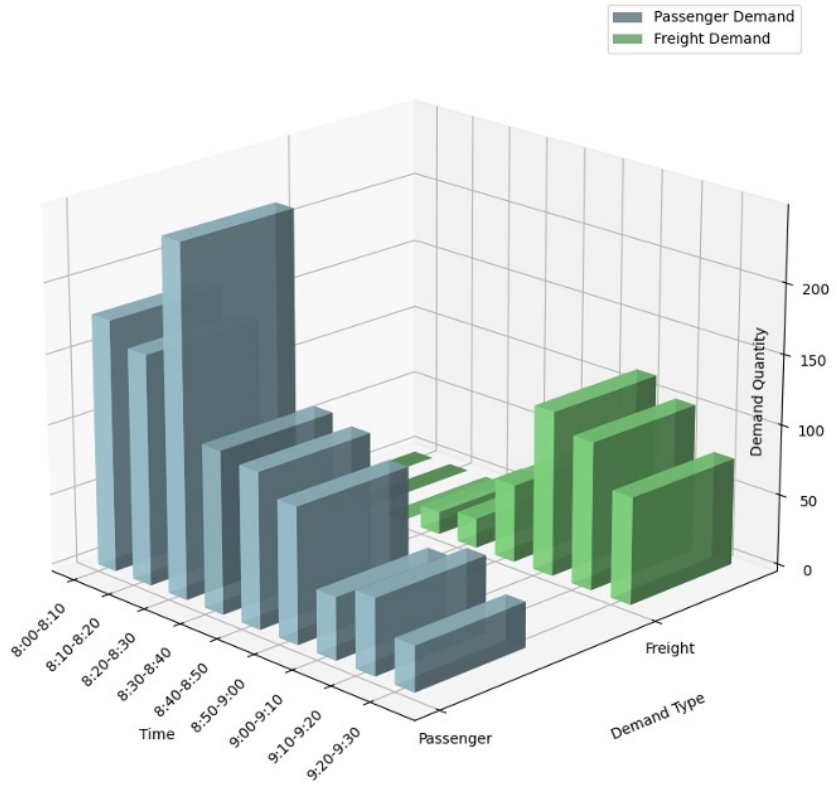

Figure 28 Distribution of demand over time

If a co-transportation transit system is not used and a strategy of transporting passengers and freight separately is adopted instead, 38 MAUs are required, with the specific path assigned to each MAU detailed in Appendix B. Since the passenger demand on an arc during off-peak periods typically does not exceed 10 people, a significant amount of idle capacity arises. This capacity is not fully utilized by freight, leading to an increase in the number of vehicles used and a corresponding rise in operational costs.

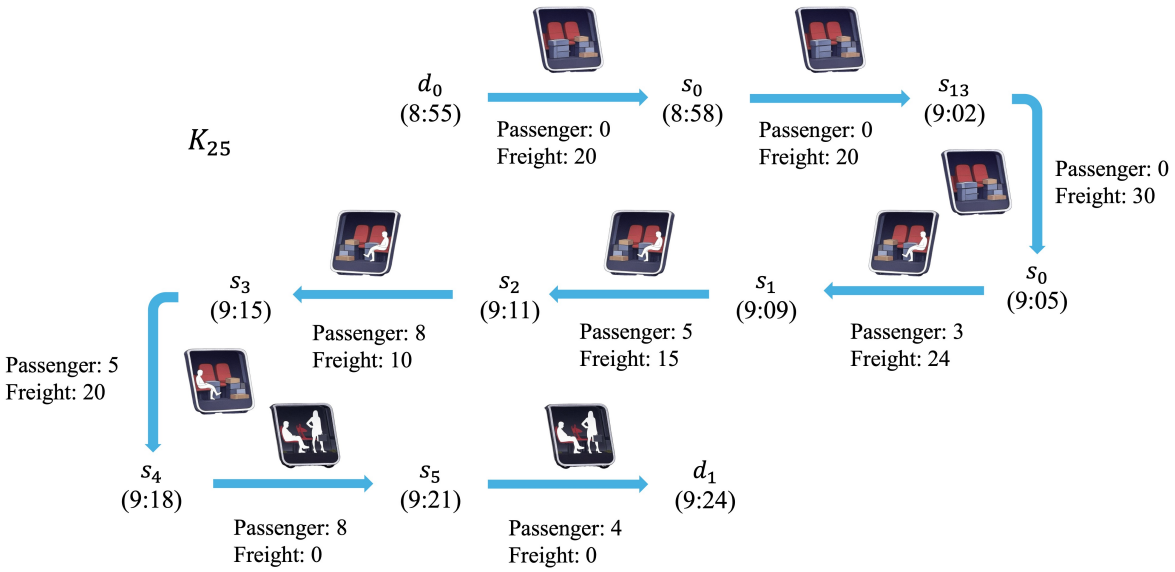

Figure 29 Example of MAU flexible transport

The electricity cost for one MAU in China is €0.036 per minute. For comparison, it is assumed that traditional buses and delivery vans are also electrically powered, with the same electricity cost of €0.089 per minute. According to the bus timetable (Bendibao 2025), a total of 5 departures are required from each of the 4 depots, with 4 departures during peak periods, each taking an average of 35 minutes to complete a transport task, and 1 departure during off-peak periods, each taking an average of 28 minutes to complete a transport task. The total time required to complete all tasks is 672 minutes. For freight demand, delivery vans are used to meet all freight requirements, departing from 4 depots to serve a total of 15 stations with freight demand. The solution is obtained using the Traveling Salesman Problem (TSP) model, with the objective of minimizing the travel time through all demand points, where the travel time is based on the average transit time of each arc.

$$\min \quad \sum_{i=0}^{14} \sum_{j=0, j\neq i}^{14} t_{ij} x_{ij} \tag{8a}$$

$$\text{s.t.} \quad \sum_{j=0, j\neq i}^{14} x_{ij} = 1 \quad \forall i \in \{0,1,\ldots,14\} \tag{8b}$$

$$\sum_{i=0, i\neq j}^{14} x_{ij} = 1 \quad \forall j \in \{0,1,\ldots,14\} \tag{8c}$$

$$u_i - u_j + 15 x_{ij} \leq 14 \quad \forall i, j \in \{1,2,\ldots,14\}, i \neq j \tag{8d}$$

$$x_{ij} \in \{0,1\} \quad \forall i, j \in \{0,1,\ldots,14\}, i \neq j \tag{8e}$$

$$u_i \in [1,14] \quad \forall i \in \{1,2,\ldots,14\} \tag{8f}$$

According to the solution obtained from the GUROBI solver, one delivery van departs from each of the four depots, with a total minimum time of 142 minutes to complete all freight demand. Since both traditional buses and delivery vans require drivers, the cost must include the salaries of all drivers (Shanghai Municipal Human Resources and Social Security Bureau 2025), with an average hourly wage of €6.5 per driver. As bus drivers can continue to perform return transport tasks after reaching the depot, a total of 12 bus drivers and 4 delivery van drivers are required.

Table 12 Cost Comparison between MAU and traditional bus & delivery van

Transport Method	Objective travel time (min)	Electricity cost (euro)	Labor cost (euro)	Total Operating cost (euro)
MAU co-transportation	1054	37.94	–	37.94
MAU separate transportation	1097	39.49	–	39.49
Traditional bus + delivery van	814	72.45	120.37	228.45

As shown in Table 12, the MAU co-transportation system offers significant operational cost advantages. Within this one-and-a-half-hour period case, it saves €1.52 compared to the separate transportation system and €190.48 compared to a traditional bus and delivery van system. Compared to a modular transit system without passenger and freight co-transportation, the cost difference in this case is only 4.1%. However, as freight demand and scale of instances increase, this could lead to considerable cost inefficiencies. In contrast, the cost difference with the traditional transportation system reaches 502.1%. Using MAUs not only reduces labor costs but also leverages the smaller capacity of each MAU compartment. During peak periods, MAUs transport passengers, and during off-peak periods, when passenger demand on an arc is low, they can utilize idle capacity to transport freight, significantly reducing capacity waste. Although the total travel time of the traditional transportation system is shorter, the larger vehicle size results in higher unit electricity consumption, leading to electricity costs nearly double those of the MAU system.

8. Conclusion

This study proposes a passenger and freight co-transportation system utilizing Modular Autonomous Units (MAUs) to fulfill all transportation demands on a network with minimal electricity consumption operational costs. MAUs can couple/decouple on any route segment, providing passenger and freight transportation services on multiple fixed bus lines (FRT), as well as freight transportation services between pick-up and delivery stops generated by daily express demands that are not on fixed lines (DRT). Through transfer arcs and delivery arcs, each MAU can move between different lines. A network is established using Space and Time dimensions, where each passenger demand has precise station and time information, and each freight demand is assigned a set of time windows. Based on this, a path-based model for the MAU Routing Problem is developed to satisfy every demand task in the network. A customized Adaptive Large Neighborhood Search (ALNS) algorithm, dividing the destruction and repair processes into path-level and arc-level, is employed to tackle the computational challenges of this complex network problem.

Due to the expansion of the overall network in both spatial and temporal dimensions, the demand quantity also increases accordingly. This study provides two methods for generating initial solutions—GUROBI and Greedy heuristic—which, when combined with the ALNS algorithm, can significantly reduce the computation time required to obtain the optimal solution. In small-scale experiments, compared to the exact optimal solution obtained by the GUROBI solver, the solution quality of both methods is proven to be within an acceptable range. The GUROBI-based algorithm deviates by no more than 1.5%, while the Greedy heuristic algorithm, despite greatly improving computation speed, deviates by no more than 5%. Computational results based on the Shanghai regional network demonstrate the advantages of the ALNS algorithm. Compared to traditional buses and delivery vans, MAUs can efficiently utilize vehicle space to transport both passengers and freight simultaneously, significantly reducing the additional operational costs caused by idle capacity within compartments. Without using a co-transportation system, even when employing MAUs for transportation, an additional 4.1% operational cost is incurred. In contrast, compared to traditional transportation systems, operational costs are reduced by 502.1%. Therefore, the use of this co-transportation modular transit system provides operators with an opportunity to reduce operational costs while meeting all transportation demands.

Overall, these findings confirm that the use of a passenger and freight co-transportation modular transit system enables operators to significantly reduce operational costs and represents a worthwhile urban internal transportation system to replace traditional public transit and delivery systems. To achieve more efficient MAU allocation on the Space-Time Network, operators should encourage close cross-functional collaboration between MAU scheduling and demand forecasting teams, supported by the robust heuristic methods for large-scale instances proposed in this study,

ensuring that FRT and DRT services mutually reinforce each other. The insights and methods proposed in this research are also broadly applicable to similar vehicle allocation scenarios combining fixed-route and demand-responsive services, providing a more flexible and sustainable methodological platform for the emerging urban transportation field.

In future research, an interesting direction is to introduce randomly occurring passenger and freight demands, enabling MAUs to further function similarly to taxis. Another promising avenue is to develop more efficient algorithms to handle larger-scale instances with more stations, timestamps, and demands, such as through constraint dualization, using column generation and row generation algorithms.

Appendix A: Detailed MAU path distribution of case study for co-transportation system

In the case study from 8:00 to 9:30 in the morning in Changning District, Shanghai, China, by using passenger and freight co-transportation system, a total of 30 MAUs are used. Table 14 shows the routes assigned to all MAUs:

Table 13: MAU path distribution of case study

Vehicle ID	Path
k_0	$d_0(8:00) > s_0(8:04) > s_1(8:08) > s_2(8:15) > s_7(8:19) > s_8(8:25) > s_9(8:29) > s_{10}(8:33) > d_3(8:38)$
k_1	$d_1(8:00) > s_5(8:05) > s_4(8:12) > s_3(8:19) > s_2(8:25) > s_7(8:30) > s_8(8:35) > s_{14}(8:39) > s_{13}(8:45) > s_0(8:52) > s_1(8:56) > s_2(9:00) > s_3(9:03) > s_{12}(9:06) > s_{11}(9:08) > s_9(9:11) > s_8(9:15) > s_7(9:18) > s_6(9:21) > d_2(9:25)$
k_2	$d_2(8:00) > s_6(8:07) > s_7(8:11) > s_8(8:15) > s_9(8:20) > s_{10}(8:27) > d_3(8:31)$
k_3	$d_3(8:00) > s_{10}(8:04) > s_9(8:11) > s_8(8:16) > s_7(8:23) > s_2(8:30) > s_1(8:36) > s_0(8:43) > d_0(8:48)$
k_4	$d_0(8:15) > s_0(8:19) > s_1(8:26) > s_2(8:32) > s_7(8:36) > s_8(8:40) > s_9(8:47) > s_{10}(8:51) > d_3(8:56)$
k_5	$d_1(8:15) > s_5(8:20) > s_4(8:26) > s_3(8:33) > s_2(8:38) > s_7(8:44) > s_8(8:48) > s_9(8:55) > s_{10}(8:59) > d_3(9:04)$
k_6	$d_1(8:15) > s_5(8:22) > s_4(8:28) > s_3(8:35) > s_2(8:41) > s_1(8:46) > s_0(8:49) > d_0(8:53)$
k_7	$d_2(8:15) > s_6(8:20) > s_7(8:25) > s_2(8:30) > s_3(8:35) > s_4(8:39) > s_5(8:45) > d_1(8:50)$
k_8	$d_3(8:15) > s_{10}(8:21) > s_9(8:27) > s_8(8:31) > s_7(8:37) > s_6(8:44) > d_2(8:49)$
k_9	$d_0(8:30) > s_0(8:35) > s_1(8:41) > s_2(8:48) > s_3(8:54) > s_4(8:59) > s_5(9:03) > d_1(9:08)$
k_{10}	$d_1(8:30) > s_5(8:35) > s_4(8:41) > s_3(8:47) > s_2(8:54) > s_1(8:56) > s_0(9:01) > d_0(9:04)$
k_{11}	$d_2(8:30) > s_6(8:37) > s_7(8:43) > s_8(8:48) > s_9(8:53) > s_{10}(8:59) > d_3(9:03)$
k_{12}	$d_3(8:30) > s_{10}(8:34) > s_9(8:38) > s_8(8:45) > s_7(8:52) > s_2(8:57) > s_1(9:02) > s_0(9:05) > d_0(9:08)$
k_{13}	$d_3(8:32) > s_{10}(8:37) > s_9(8:43) > s_8(8:49) > s_7(8:55) > s_6(9:00) > d_2(9:04)$
k_{14}	$d_0(8:35) > s_0(8:40) > s_1(8:47) > s_2(8:53) > s_7(8:58) > s_6(9:02) > d_2(9:07)$
k_{15}	$d_3(8:38) > s_{10}(8:42) > s_9(8:48) > s_8(8:54) > s_7(9:01) > s_2(9:05) > s_1(9:08) > s_0(9:11) > d_0(9:15)$
k_{16}	$d_3(8:42) > s_{10}(8:47) > s_9(8:52) > s_8(8:59) > s_7(9:03) > s_2(9:06) > s_3(9:10) > s_4(9:13) > s_5(9:17) > d_1(9:21)$
k_{17}	$d_0(8:45) > s_0(8:49) > s_1(8:53) > s_2(8:58) > s_3(9:03) > s_4(9:07) > s_5(9:11) > d_1(9:15)$
k_{18}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{19}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{20}	$d_3(8:45) > s_{10}(8:52) > s_9(8:56) > s_8(9:03) > s_7(9:08) > s_6(9:10) > d_2(9:13)$

Continued on next page

Table 13: MAU path distribution of case study

Vehicle ID	Path
k_{21}	$d_0(8:48) > s_0(8:53) > s_1(8:59) > s_2(9:05) > s_7(9:09) > s_6(9:13) > d_2(9:16)$
k_{22}	$d_3(8:48) > s_{10}(8:53) > s_9(8:59) > s_8(9:04) > s_7(9:08) > s_2(9:12) > s_1(9:15) > s_0(9:19) > d_0(9:23)$
k_{23}	$d_2(8:50) > s_6(8:54) > s_7(8:58) > s_8(9:03) > s_9(9:07) > s_{10}(9:11) > d_3(9:15)$
k_{24}	$d_0(8:50) > s_0(8:54) > s_1(8:58) > s_2(9:03) > s_3(9:07) > s_4(9:11) > s_5(9:15) > d_1(9:19)$
k_{25}	$d_0(8:55) > s_0(8:58) > s_{13}(9:02) > s_0(9:05) > s_1(9:09) > s_2(9:11) > s_3(9:15) > s_4(9:18) > s_5(9:21) > d_1(9:24)$
k_{26}	$d_0(8:55) > s_0(8:59) > s_1(9:03) > s_2(9:07) > s_3(9:11) > s_4(9:15) > s_5(9:19) > d_1(9:23)$
k_{27}	$d_1(9:00) > s_5(9:06) > s_4(9:10) > s_3(9:15) > s_2(9:19) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{28}	$d_2(9:00) > s_6(9:07) > s_7(9:11) > s_8(9:15) > s_9(9:18) > s_{10}(9:23) > d_3(9:26)$
k_{29}	$d_3(9:00) > s_9(9:05) > s_{11}(9:09) > s_8(9:13) > s_{14}(9:16) > s_1(9:19) > s_0(9:22) > d_0(9:25)$

Appendix B: Detailed MAU path distribution of case study for separate transportation system

In the case study from 8:00 to 9:30 in the morning in Changning District, Shanghai, China, by using passenger and freight separate system, a total of 30 MAUs are used. Table 14 shows the routes assigned to all MAUs:

Table 14: MAU path distribution for passenger and freight demand

Vehicle ID	Path
MAU for passenger demand:	
k_0	$d_0(8:00) > s_0(8:04) > s_1(8:08) > s_2(8:15) > s_7(8:19) > s_8(8:25) > s_9(8:29) > s_{10}(8:33) > d_3(8:38)$
k_1	$d_1(8:00) > s_5(8:05) > s_4(8:12) > s_3(8:19) > s_2(8:25) > s_7(8:31) > s_6(8:36) > d_2(8:41)$
k_2	$d_2(8:00) > s_6(8:07) > s_7(8:11) > s_8(8:15) > s_9(8:20) > s_{10}(8:27) > d_3(8:31)$
k_3	$d_3(8:00) > s_{10}(8:04) > s_9(8:11) > s_8(8:16) > s_7(8:23) > s_2(8:30) > s_1(8:36) > s_0(8:43) > d_0(8:48)$
k_4	$d_0(8:15) > s_0(8:19) > s_1(8:26) > s_2(8:32) > s_7(8:36) > s_8(8:40) > s_9(8:47) > s_{10}(8:51) > d_3(8:56)$
k_5	$d_1(8:15) > s_5(8:20) > s_4(8:26) > s_3(8:33) > s_2(8:38) > s_7(8:44) > s_8(8:48) > s_9(8:55) > s_{10}(8:59) > d_3(9:04)$
k_6	$d_1(8:15) > s_5(8:22) > s_4(8:28) > s_3(8:35) > s_2(8:41) > s_1(8:46) > s_0(8:49) > d_0(8:53)$
k_7	$d_2(8:15) > s_6(8:20) > s_7(8:25) > s_2(8:30) > s_3(8:35) > s_4(8:39) > s_5(8:45) > d_1(8:50)$
k_8	$d_3(8:15) > s_{10}(8:21) > s_9(8:27) > s_8(8:31) > s_7(8:37) > s_6(8:44) > d_2(8:49)$
k_9	$d_0(8:30) > s_0(8:35) > s_1(8:41) > s_2(8:48) > s_3(8:54) > s_4(8:59) > s_5(9:03) > d_1(9:08)$

Continued on next page

Table 14: MAU path distribution for passenger and freight demand

Vehicle ID	Path
k_{10}	$d_1(8:30) > s_5(8:35) > s_4(8:41) > s_3(8:47) > s_2(8:54) > s_1(8:56) > s_0(9:01) > d_0(9:04)$
k_{11}	$d_2(8:30) > s_6(8:37) > s_7(8:43) > s_8(8:48) > s_9(8:53) > s_{10}(8:59) > d_3(9:03)$
k_{12}	$d_3(8:30) > s_{10}(8:34) > s_9(8:38) > s_8(8:45) > s_7(8:52) > s_2(8:57) > s_1(9:02) > s_0(9:05) > d_0(9:08)$
k_{13}	$d_3(8:38) > s_{10}(8:42) > s_9(8:48) > s_8(8:54) > s_7(9:01) > s_2(9:05) > s_1(9:08) > s_0(9:11) > d_0(9:15)$
k_{14}	$d_3(8:42) > s_{10}(8:47) > s_9(8:52) > s_8(8:59) > s_7(9:03) > s_2(9:06) > s_3(9:10) > s_4(9:13) > s_5(9:17) > d_1(9:21)$
k_{15}	$d_0(8:45) > s_0(8:49) > s_1(8:53) > s_2(8:58) > s_3(9:03) > s_4(9:07) > s_5(9:11) > d_1(9:15)$
k_{16}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{17}	$d_1(8:45) > s_5(8:50) > s_4(8:54) > s_3(8:59) > s_2(9:04) > s_1(9:09) > s_0(9:11) > d_0(9:14)$
k_{18}	$d_3(8:45) > s_{10}(8:52) > s_9(8:56) > s_8(9:03) > s_7(9:08) > s_6(9:10) > d_2(9:13)$
k_{19}	$d_0(8:48) > s_0(8:53) > s_1(8:59) > s_2(9:05) > s_7(9:09) > s_6(9:13) > d_2(9:16)$
k_{20}	$d_3(8:48) > s_{10}(8:53) > s_9(8:59) > s_8(9:04) > s_7(9:08) > s_2(9:12) > s_1(9:15) > s_0(9:19) > d_0(9:23)$
k_{21}	$d_2(8:50) > s_6(8:54) > s_7(8:58) > s_8(9:03) > s_9(9:07) > s_{10}(9:11) > d_3(9:15)$
k_{22}	$d_0(8:50) > s_0(8:54) > s_1(8:58) > s_2(9:03) > s_3(9:07) > s_4(9:11) > s_5(9:15) > d_1(9:19)$
k_{23}	$d_0(8:55) > s_0(8:58) > s_1(9:03) > s_2(9:06) > s_3(9:10) > s_4(9:14) > s_5(9:18) > d_1(9:21)$
k_{24}	$d_0(8:55) > s_0(8:59) > s_1(9:03) > s_2(9:07) > s_3(9:11) > s_4(9:15) > s_5(9:19) > d_1(9:23)$
k_{25}	$d_1(9:00) > s_5(9:06) > s_4(9:10) > s_3(9:15) > s_2(9:19) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{26}	$d_2(9:00) > s_6(9:07) > s_7(9:11) > s_8(9:15) > s_9(9:18) > s_{10}(9:23) > d_3(9:26)$
k_{27}	$d_3(9:00) > s_9(9:05) > s_{11}(9:09) > s_8(9:13) > s_{14}(9:16) > s_1(9:19) > s_0(9:22) > d_0(9:25)$
MAU for freight demand:	
k_{28}	$d_0(8:55) > s_0(8:58) > s_{13}(9:02) > s_0(9:05) > s_1(9:09) > s_2(9:11) > s_3(9:15) > s_4(9:18) > s_5(9:21) > d_1(9:24)$
k_{29}	$d_1(8:58) > s_5(9:03) > s_4(9:04) > s_{12}(9:06) > s_{11}(9:08) > s_9(9:11) > s_8(9:15) > s_7(9:18) > s_6(9:21) > d_2(9:25)$
k_{30}	$d_2(9:00) > s_6(9:04) > s_7(9:07) > s_8(9:10) > s_{12}(9:15) > s_3(9:19) > s_4(9:22) > s_5(9:25) > d_1(9:28)$
k_{31}	$d_1(9:00) > s_5(9:06) > s_4(9:10) > s_3(9:15) > s_2(9:19) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{32}	$d_2(9:00) > s_6(9:07) > s_7(9:11) > s_8(9:15) > s_9(9:18) > s_{10}(9:23) > d_3(9:26)$
k_{33}	$d_3(9:00) > s_9(9:05) > s_{11}(9:09) > s_8(9:13) > s_{14}(9:16) > s_1(9:19) > s_0(9:22) > d_0(9:25)$
k_{34}	$d_3(9:02) > s_{10}(9:06) > s_9(9:10) > s_8(9:13) > s_7(9:17) > s_6(9:20) > d_2(9:24)$

Continued on next page

Table 14: MAU path distribution for passenger and freight demand

Vehicle ID	Path
k_{35}	$d_3(9:02) > s_{10}(9:06) > s_9(9:10) > s_8(9:13) > s_7(9:17) > s_2(9:20) > s_1(9:23) > s_0(9:27) > d_0(9:30)$
k_{36}	$d_2(9:03) > s_6(9:06) > s_7(9:09) > s_8(9:12) > s_9(9:15) > s_{10}(9:20) > d_3(9:23)$
k_{37}	$d_0(9:05) > s_0(9:08) > s_1(9:11) > s_{13}(9:14) > s_{14}(9:18) > s_7(9:21) > s_6(9:24) > d_2(9:27)$

References

Bendibao, 2025 *Shanghai bus query*. URL `https://m.sh.bendibao.com/bus/`, accessed: 2025-07-26.

EV Connect, 2025 *Ev charging costs at public stations*. URL `https://www.evconnect.com/blog/ev-charging-costs-at-public-stations`, accessed: 2025-07-22.

Hassold S, Ceder AA, 2014 *Public transport vehicle scheduling featuring multiple vehicle types. Transportation Research Part B: Methodological* 67:129–143.

Hatzenbühler J, Jenelius E, Gidófalvi G, Cats O, 2023 *Modular vehicle routing for combined passenger and freight transport. Transportation Research Part A: Policy and Practice* 173:103688.

Hatzenbühler J, Jenelius E, Gidófalvi G, Cats O, 2024 *Multi-purpose pickup and delivery problem for combined passenger and freight transport. Transportation* 1–32.

Holguín-Veras J, Amaral JC, Rivera-Gonzalez C, 2024 *Using gps data to assess the effectiveness of freight demand management strategies. Transportation Research Procedia* 79:21–28.

Li L, Negenborn RR, De Schutter B, 2013 *A general framework for modeling intermodal transport networks. 2013 10th IEEE International Conference on Networking, Sensing and Control (ICNSC)*, 579–585 (IEEE).

Li S, Zhu X, Shang P, Li T, Liu W, 2023 *Optimizing a shared freight and passenger high-speed railway system: A multi-commodity flow formulation with benders decomposition solution approach. Transportation Research Part B* 172:1–31.

Lin J, Nie YM, Kawamura K, 2022 *An autonomous modular mobility paradigm. IEEE Intelligent Transportation Systems Magazine* 15(1):378–386.

Lin J, Zhang F, 2024 *Modular vehicle-based transit system for passenger and freight co-modal transportation. Transportation Research Part C* 160.

Liu X, Qu X, Ma X, 2021 *Improving flex-route transit services with modular autonomous vehicles. Transportation Research Part E* 149.

Machado B, Pimentel C, de Sousa A, 2023 *Integration planning of freight deliveries into passenger bus networks: Exact and heuristic algorithms. Transportation Research Part A: Policy and Practice* 171:103645.

Mason WF, 1967 *Air freight. Financial Analysts Journal* 23(5):49–56.

NExT Modular Vehicles, 2024 *Smart urban transportation*. URL `https://www.next-future-mobility.com`, modular electric vehicles for sustainable urban mobility.

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

Shanghai Municipal Human Resources and Social Security Bureau, 2025 *Shanghai human resources and social security bureau website*. URL `https://rsj.sh.gov.cn`, accessed: 2025-07-26.

Shaw P, 1997 *A new local search algorithm providing high quality solutions to vehicle routing problems*. Technical report, APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK.

Shi X, Li X, 2021 *Operations design of modular vehicles on an oversaturated corridor with first-in, first-out passenger queueing. Transportation Science* 55(5):1187–1205.

Solar Impulse Foundation, 2025 *Solutions explorer*. URL `https://solarimpulse.com/solutions-explorer`, accessed: 2025-07-22.

Tang C, Liu J, Ceder A, Jiang Y, 2024 *Optimisation of a new hybrid transit service with modular autonomous vehicles. Transportmetrica A: Transport Science* 20(2).

Xia D, Ma J, Sharif Azadeh S, 2024a *Integrated timetabling and vehicle scheduling of an intermodal urban transit network: A distributionally robust optimization approach. Transportation Research Part C: Emerging Technologies* 162:104610.

Xia D, Ma J, Sharif Azadeh S, 2024b *Integrated timetabling, vehicle scheduling, and dynamic capacity allocation of modular autonomous vehicles under demand uncertainty*. arXiv:2410.16409.

Xia D, Ma J, Sharif Azadeh S, Zhang W, 2023 *Data-driven distributionally robust timetabling and dynamic-capacity allocation for automated bus systems with modular vehicles. Transportation Research Part C: Emerging Technologies* 155:104314.

Yang Y, Xie X, Wang Z, 2024 *Optimization of high-speed rail express transportation scheduling based on spatio-temporal-state network modeling. Applied Sciences* 14(22):10456.

Zeng Z, Qu X, 2022 *Optimization of electric bus scheduling for mixed passenger and freight flow in an urban-rural transit system. IEEE Transactions on Intelligent Transportation Systems* 24(1):1288–1298.

Zhou C, Dai P, Wang F, Zhang Z, 2016 *Predicting the passenger demand on bus services for mobile users. Pervasive and Mobile Computing* 25:48–66.

Zhu S, Bell MG, Schulz V, Stokoe M, 2023 *Co-modality in city logistics: Sounds good, but how? Transportation Research Part A: Policy and Practice* 168:103578.