systems

MDPI

Article

Combining MAMBA and Attention-Based Neural Network for Electric Ground-Handling Vehicles Scheduling

Jiawei Li †, Weigang Fu *,†, Gangjin Huang *, Kai Liu, Jiewei Zhang and Yaoming Fu

Aviation Engineering Institute, Civil Aviation Flight University of China, Guanghan 618307, China
* Correspondence: jiaodafwg@126.com (W.F.); huanggj@cafuc.edu.cn (G.H.)
† These authors contributed equally to this work.

Abstract: To reduce airport operational costs and minimize environmental pollution, an increasing number of airports are transitioning from fuel-powered to electric ground-handling vehicles. However, the limited battery capacity of electric vehicles and the need for charging make the scheduling of these vehicles more complex. To address this scheduling problem, this paper proposes an electric ground-handling vehicle scheduling algorithm that combines the MAMBA model with an attention-based neural network. The MAMBA model is designed to process multi-dimensional features such as flight information, vehicle locations, service demands, and time window constraints. Subsequently, an attention mechanism-based neural network is developed to dynamically integrate vehicle states, service records, and operational and charging constraints, in order to select the most suitable flights for electric ground-handling vehicles to service. The experiments use flight data from Xiamen Gaoqi International Airport and compare the proposed method with CPLEX solvers, existing heuristic algorithms, and custom heuristic algorithms. The results demonstrate that the proposed method not only effectively solves the electric ground-handling vehicle scheduling problem and provides high-quality solutions, but also exhibits good scalability in different parameter settings and real-time scheduling scenarios.

Keywords: electric ground-handling vehicle; airport operations; vehicle scheduling; deep learning; MAMBA model; attention model; reinforcement learning; MILP

check for updates

Academic Editor: Mahyar Amirgholy

Received: 17 January 2025
Revised: 9 February 2025
Accepted: 18 February 2025
Published: 26 February 2025

Citation: Li, J.; Fu, W.; Huang, G.; Liu, K.; Zhang, J.; Fu, Y. Combining MAMBA and Attention-Based Neural Network for Electric Ground-Handling Vehicles Scheduling. *Systems* **2025**, *13*, 155. https://doi.org/10.3390/systems13030155

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Efficient airport operations rely heavily on the support of ground-handling vehicles, which play a crucial role in ensuring flight safety, enhancing operational efficiency, and meeting passenger demands. Based on practical research at airports, the primary process of ground support services can be summarised as illustrated in Figure 1. As awareness of environmental issues and sustainability grows, many airports are transitioning to electric ground-handling vehicles. However, this shift introduces significant challenges in scheduling, as the scheduling of electric vehicles has not been extensively researched, and traditional models fail to address the complexities of modern airport operations. Consequently, optimizing the scheduling of electric ground-handling vehicles has become a critical concern for airport management.

To ensure smooth airport operations, it is essential to develop scheduling solutions tailored to different flight types. However, most airports still rely on manual scheduling for single-vehicle, single-flight service [1], which is inefficient, particularly with electric vehicles' complex needs. Manual scheduling leads to low vehicle utilization, difficulties in handling emergencies, and an increased risk of flight delays. Therefore, there is a pressing

need for an intelligent and efficient scheduling method that not only capitalizes on the advantages of electric vehicles but also addresses the inefficiencies of current manual scheduling approaches.

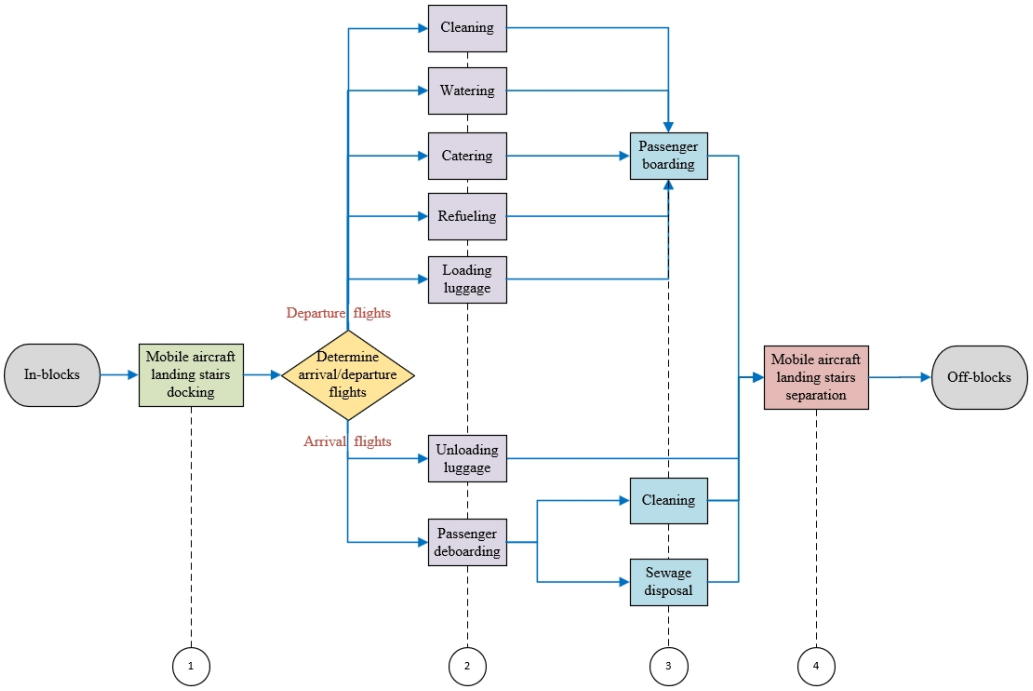


Figure 1. The primary service processes of ground-handling vehicles.

To solve the scheduling problem of electric ground-handling vehicles at airports, this study proposes an innovative algorithm that combines the Multi-scale Adaptive Memory-Based Attention (Mamba) model with an attention-based neural network. MAMBA is a neural network architecture based on State-Space Models (SSM), capable of efficiently modeling long-range dependencies in sequential data, making it particularly suitable for time series prediction and optimization problems. SSM is a mathematical framework used to model time series data, capable of describing the dynamic evolution of a system. It has significant applications in signal processing, control systems, and machine learning. A key challenge in this problem is how to effectively formulate the scheduling process as an optimization model that accounts for battery levels, charging requirements, and service constraints. To address this, the Mamba model is employed to process multi-dimensional features, including the positions, service requirements, and time windows of flights and vehicles. Through its SSM mechanism, the model captures complex relationships between scheduling elements, enabling more informed decision-making. Furthermore, selecting the most suitable flights for electric ground-handling vehicles while considering real-time operational constraints remains a critical issue. To tackle this, an attention-based neural network is integrated into the scheduling framework to dynamically allocate vehicles to flights, ensuring both efficiency and feasibility. Reinforcement learning (RL) is incorporated to continuously optimize the scheduling strategy, enhancing solution quality and improving the overall safety and efficiency of airport operations. Accordingly, this study contributions are summarized as follows:

(1) This study formulates the scheduling problem of electric ground-handling vehicles as a mixed-integer programming model, explicitly considering constraints such as battery levels and charging requirements. The formulation establishes a structured optimization framework that simplifies the problem while enhancing computational efficiency.

(2) This study develops a novel scheduling framework based on the Mamba model, which is specifically designed to handle the electric ground-handling vehicle scheduling. By leveraging the SSM mechanism, this framework enables the Mamba model to comprehensively process multi-dimensional scheduling features, including flight locations, vehicle positions, service requirements, and time windows. The proposed method effectively captures complex relationships between nodes, significantly improving the encoding quality and subsequent decision-making.

(3) This study proposes a customized attention-based neural network tailored for electric ground-handling vehicle scheduling. Unlike existing methods, this network is explicitly designed to dynamically select flights while accounting for charging constraints. Furthermore, reinforcement learning is integrated into the framework to optimize scheduling strategies, allowing the algorithm to autonomously generate efficient and high-quality scheduling solutions.

The remaining sections of this paper are structured as follows. Section 2 reviews existing work on ground-handling vehicle scheduling, Vehicle Routing Problem (VRP), and their variants. Section 3 outlines the problem of scheduling electric ground-handling vehicles, basic assumptions, and establishes a model for scheduling these vehicles. Section 4 provides a detailed explanation of the designed algorithm. Section 5 presents experimental results from tests on various algorithms using flight data from Xiamen Gaoqi Airport and an analysis of the outcomes. Section 6 concludes the paper.

2. Related Works

Compared to other airport resource management, the ground-handling vehicle scheduling problem has received less attention, with relatively scarce related research. Moreover, while most existing studies focus on fuel-powered ground-handling vehicles, there has been no in-depth discussion on the scheduling problem of electric ground-handling vehicles. Therefore, this research proposed intelligent scheduling algorithm combining the Mamba model with attention neural network aims to fill this research gap and improve the scheduling efficiency and operational benefits of electric ground-handling vehicles. Most existing research on the scheduling of ground-handling vehicles models a single operation, typically treating it as a VRP or its variants. For example, Wang and his team focused on scheduling airport refueling trucks and used the Particle Swarm Optimization (PSO) algorithm to solve this problem [2]. Han et al. formulated the ferry vehicle scheduling problem as an Mixed-Integer Linear Programming (MILP) problem to minimize the number of vehicles used, solving it through Convex Programming Linear Expression (CPLEX) [3]. MILP is a mathematical optimization technique that involves linear relationships with some variables restricted to integer values, aiming to find the best solution under a set of constraints. CPLEX is a powerful mathematical optimization solver. It is extensively applied in operational optimization and scheduling fields. Guo et al. proposed an enhancement genetic algorithm that incorporates a cooperative game mechanism, validated by airport baggage transport vehicles, which outperformed other algorithms [4]. Feng et al. proposed a collaborative scheduling method for electric ground-handling vehicles based on a Simple Temporal Network (STN), achieving independent scheduling across departments [5]. Zampirolli and Amaral used Simulated Annealing (SA) and Iterated Local Search (ILS) algorithms to solve the problem of aircraft refueling [6]. Han et al. proposed a two-stage optimization model to enhance the scheduling efficiency of airport ferry vehicles and reduce service delay times, with the first stage focusing on optimizing on-time service for flights and the second stage addressing the delay of unserved flights [7].

Only a few studies have focused on the overall optimization of various operations in the scheduling of ground-handling vehicles. To address these complex problems, re-

searchers have developed specific metaheuristic algorithms. Liu et al. proposed a dual-objective model based on Non-dominated Sorting Genetic Algorithm 2 (NSGA2), aiming to reduce the number of special vehicles and total additional time costs. NSGA2 is a widely used multi-objective evolutionary algorithm that applies non-dominated sorting and crowding distance to solve optimization problems with multiple conflicting objectives. Its performance is superior to the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) and the PSO algorithm [8]. Zhang et al. studied the scheduling problem of airport vehicles under resource constraints, adjusted the vehicles and their schedules using a sequential sorting method, and solved the problem using a multi-layer coding genetic algorithm [9]. To reduce waiting times and total completion times for flight services, Padrón and Guimarans optimized the time window settings for the fleet using Constraint Programming (CP), then used the Large Neighborhood Search (LNS) algorithm to solve the decomposed Vehicle Routing Problem with Time Windows (VRPTW) problem for each fleet [10]. Zhou et al. developed a learning-assisted large neighborhood search method that integrates imitation learning and graph convolutional networks to optimize airport ground-handling vehicle fleet scheduling, effectively managing up to 200 flights with 10 types of operations and outperforming existing methods [11].

Metaheuristic algorithms have shown great potential in solving VRPs and their various variants, especially in scheduling electric ground-handling vehicles. For example, Zhao and his team constructed a multi-objective optimization model covering cost, carbon emissions, and customer satisfaction, and designed an improved Multi-objective Ant Colony Optimization (MOACO) algorithm incorporating multi-objective heuristic functions to address this challenge [12]. Attar et al. focused on the production allocation problem of electric vehicles, considering its characteristics and constraints, constructed a MILP model, and designed two efficient and accurate algorithms [13]. Breunig et al. addressed the electric two-stage VRP, solving small-scale instances with exact algorithms while developing large-scale neighborhood search algorithms to tackle large instances [14]. Ganji, Kazemipoor, and Molana compared the performance of three multi-objective metaheuristic algorithms: Multi-Objective Particle Swarm Optimization (MOPSO), NSGA2, and MOACO in vehicle route optimization. The experimental results showed that NSGA2 outperformed the others in terms of solution quality and processing efficiency [15]. Xiong developed an Improved Ant Colony Optimization (IACO) algorithm, using a mixed pheromone update strategy that demonstrated significant advantages over traditional ant colony algorithms in optimizing logistics distribution paths [16]. Wang et al. proposed a distributed strategy for tractor services, developed an improved NSGA2 algorithm, and successfully reduced latency time [17]. Han et al. introduced a dual-objective MILP model and designed three heuristic algorithms to improve the efficiency and balanced utilization of ferry scheduling [18].

Recently, deep learning algorithms have made significant breakthroughs in solving various variants of the VRP, such as the VRPTW and so on. Most of these deep learning models adopt an encoder-decoder structure and are trained using RL algorithms. Researchers have developed frameworks for using RL to solve combinatorial optimization problems. Xu et al. presented a novel attention-based RL model that enhances node embeddings and context embedding through batch normalization reordering, gate aggregation, and an attentive aggregation module, demonstrating superior performance and faster problem-solving across various vehicle routing problems compared to traditional and learning-based methods [19]. Xin et al. introduced a novel step-wise deep learning approach for routing problems in intelligent transportation systems, significantly enhancing the performance of the pointer network and Transformer Attention Model (TAM) by explicitly removing visited nodes in each selection step and proposing an approximate step-wise TAM model for larger

instances [20]. Lei et al. presented an end-to-end deep reinforcement learning framework that can be adapted to various combinatorial optimization problems with minimal changes, significantly improving solution quality for taveling salesman problem and capacitated vehicle routing problem with up to 100 nodes and demonstrating strong generalization capabilities [21]. Zou et al. introduced an improved transformer model (TAOA) that integrates multi-head attention mechanism (MHA) and attention-to-attention mechanism (AOA) to enhance the solution of the low-carbon multi-depot vehicle routing problem (MDVRP), demonstrating better performance compared to traditional methods [22]. Joshi, Laurent, and Bresson efficiently solved the taveling salesman problem problem using graph convolutional networks, proposed a new model, and optimized model parameters through the policy gradient method, achieving near-optimal solutions on a two-dimensional euclidean graph [23]. However, it is worth noting that most current deep learning algorithms mainly focus on classic VRP variants with simple constraints, which may limit their applicability to complex problems such as electric ground-handling vehicle scheduling.

In summary, existing research addresses the traditional ground-handling vehicle scheduling problem. This problem is typically modeled as a VRP or its variant. Various metaheuristic algorithms, such as MOEA/D and PSO, have been developed for both single and multiple vehicle scheduling. However, significant gaps remain in addressing the unique challenges posed by electric ground-handling vehicles. Although some studies have explored deep RL techniques for path optimization, they mainly focus on VRP problems and simple VRP variants, and cannot fully address the scheduling problems of electric ground-handling vehicles. Given this research gap, this paper innovatively proposes an algorithm combining the MAMBA model and attention neural networks, focusing on solving electric ground-handling vehicle scheduling problems.

3. Problem Description and Model Construction

According to airport site manuals, both departure and arrival flights at an airport require ground-handling vehicles to perform a series of service operations, such as boarding and deboarding passengers, refueling, providing meals, adding water and sewage, loading (unloading) cargo and luggage, and towing the aircraft. These services must be completed by the corresponding types of special vehicles, i.e., electric mobile aircraft landing stairs (for remote stands), electric airport shuttle buses (for remote stands), electric refueling trucks (tanker/pump), electric food delivery vehicles, electric water service vehicles, electric sewage trucks, electric waste collection vehicles, electric baggage tractors, electric baggage conveyor vehicles, and electric aircraft tow vehicles. Figure 2 illustrates the electric ground-handling vehicles required by civil aircraft when parked at an airport.

3.1. Problem Description

The primary operational cost for electric ground-handling vehicles is the energy consumed during travel. Therefore, designing scheduling solutions to minimize the vehicles' travel distances is beneficial for reducing operational costs, as well as decreasing vehicle wear and maintenance expenses. This research formulates the electric ground-handling vehicle scheduling problem as a Mixed Integer Linear Programming (MILP) model. Figure 3 illustrates the flowchart of the procedure, visually outlining the steps involved in solving the scheduling problem. It provides a clear representation of the key stages, from data input to the optimization process, and highlights the decision-making steps for minimizing the travel distances of the vehicles.

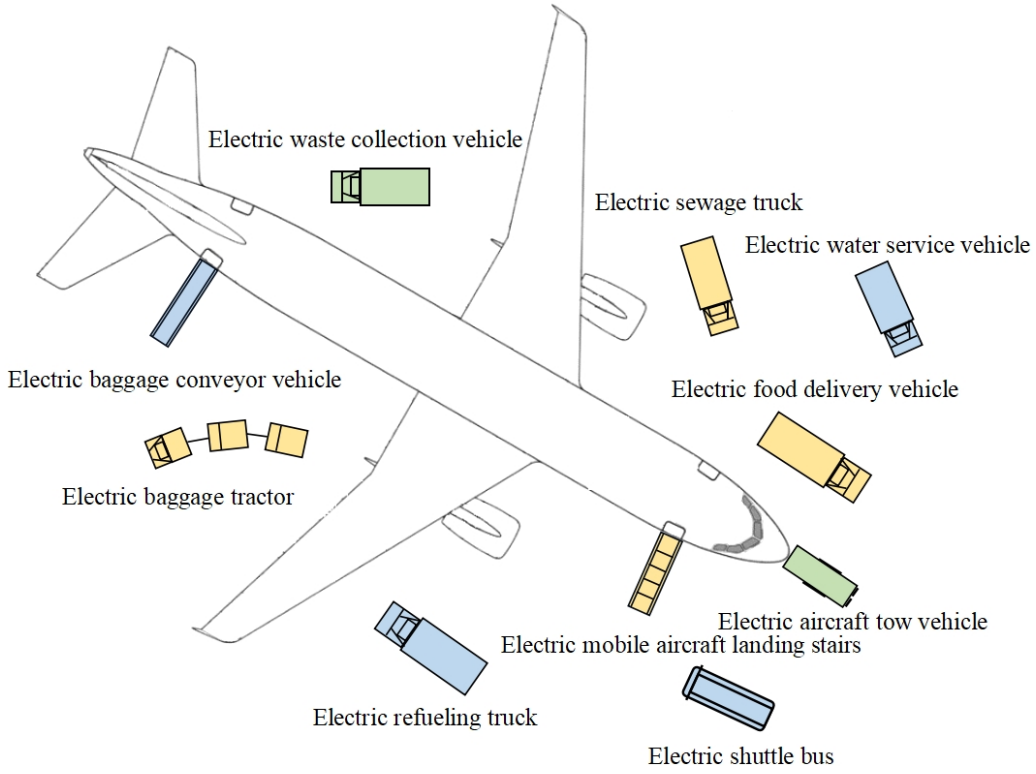


Figure 2. Electric ground-handling vehicles serving civil aircraft.

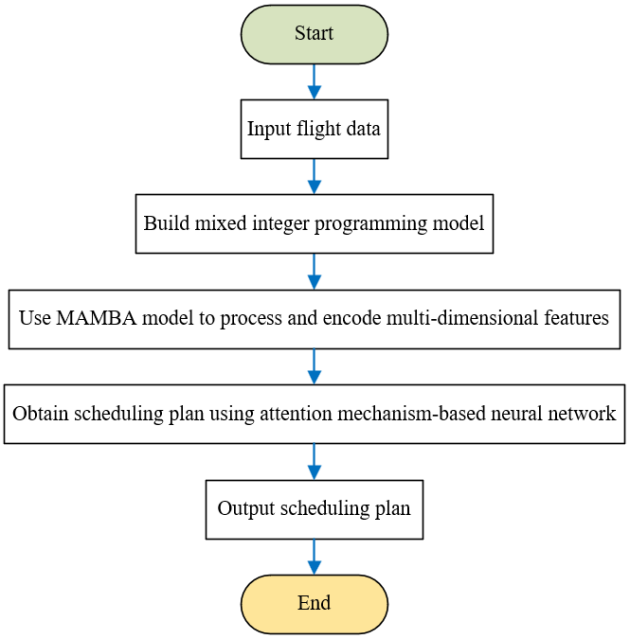


Figure 3. Flowchart of the scheduling procedure.

3.2. Basic Assumptions

In the assessment of energy consumption for electric ground-handling vehicles, several complex challenges exist. This study focuses solely on the total travel distance of electric vehicles, excluding energy consumption during service operations. Various factors influence energy consumption, including vehicle characteristics, driving behavior, predictive models, and testing standards. Differences in battery performance, motor efficiency, and other vehicle-specific characteristics lead to variability in energy consumption. Additionally, factors such as vehicle acceleration and speed are closely related to energy usage, while the complex operating conditions at airports and environmental factors further complicate

the issue [24,25]. Existing energy consumption prediction models have limitations and are not easily applicable to electric ground-handling vehicles due to the difficulty in obtaining the necessary input parameters [24]. Differences in driving styles can also significantly impact energy consumption, but precisely quantifying these effects remains difficult [25]. Furthermore, current testing standards differ significantly from actual operating conditions, making it challenging to reliably assess real-world energy consumption [26]. Considering these factors, the direct relationship between travel distance and scheduling decisions makes it relatively easier to quantify and optimize travel energy consumption. In contrast, energy consumption during service operations is influenced by many complex, hard-to-measure factors. Incorporating these factors into the model would significantly increase its complexity and compromise its accuracy and practicality. Therefore, this study excludes service energy consumption from modeling and optimization. The assumptions of the MILP model were as follows:

1. One type of electric ground-handling vehicle provides one specific service for different flights.
2. The starting point of all vehicles in the same depot.
3. The same types of electric ground-handling vehicles are identical to each other.
4. Once the service is started, it will not be interrupted until it is completed.
5. Due to a small number of vehicles and a relatively large number of flights, the vehicle will continuously provide service for flights unless its charging conditions are met, at that time point the vehicle will travel to the depot for recharging.
6. This study primarily focuses on optimizing travel routes and task allocation for better solutions. Due to the direct correlation between energy consumption during travel and scheduling decisions, while energy consumption related to services is relatively fixed and unaffected by scheduling strategies, the current scheduling solution does not consider the energy consumption of electric ground-handling vehicles during service, such as battery usage when traveling between tasks or idling during standby periods. Other types of consumption, such as fuel consumption of airplanes or overall airport energy usage, are also beyond the scope of this study.
7. The linear approximation is found to be reasonable when considering the electric vehicle charging speed [27]. This study considered the charging speed of the electric ground-handling vehicle is linear as well.

3.3. Notation Descriptions

The notations and their meanings for model are as shown in Table 1.

Table 1. Notation descriptions.

Indexes	
Set	
V	The set of electric ground-handling vehicles, where $V = \{1, 2, \ldots, n_v\}$ represents the total number of vehicles available for scheduling.
F	The set of flights, where $F = \{1, 2, \ldots, n_f\}$ denotes the total number of flights that require ground-handling services.
n_v	The total number of electric ground-handling vehicles
n_f	The total number of flights
n_{vs}	The number of electric ground-handling vehicles providing service

Table 1. *Cont.*

Indexes	
Parameters	
ST_{ij}	The total time for electric ground-handling vehicle i during service for flight j
TD_{ij}	The travel distance for electric ground-handling vehicle i during service for flight j
TT_{ij}	The travel time for electric ground-handling vehicle i serving for flight j during the total time ST_{ij}
$TD_{ij'}$	The travel distance of i serving for flight j' during the service time $ST_{ij'}$
S_i	The average travel speed of electric ground-handling vehicle i
BT_{ij}	The time at which electric ground-handling vehicle i starts being occupied by flight j
$BT_{ij'}$	The time at which electric ground-handling vehicle i starts being occupied by flight j'
$BT_{i'j}$	The time at which electric ground-handling vehicle i' with lower priority than vehicle i starts being occupied by flight j
D_{jp}	The distance from the location of flight j to the depot p
VD_i^m	The distance electric ground-handling vehicle i traveling with full capacity
VD_{ij}	The distance electric ground-handling vehicle i traveling with its current battery level
CT_i^m	The time of electric ground-handling vehicle i to charge its battery level from 0 to full capacity.
CT_i^c	The time of electric ground-handling vehicle i charged from its current battery level to full capacity
FT_i	the idle time of electric ground-handling vehicle i between serving flights j and j'
CR_i	The distance of electric ground-handling vehicle i traveling per unit charging time
D_j	The demand of flight j during the service time
Q_i	The capacity of electric ground-handling vehicle i
Binary variables	
x_{ij}	The indicator variable representing whether electric ground-handling vehicle i serves flight j
C_i	The indicator variable representing whether electric ground-handling vehicle i is charging

3.4. Model Formulation

Based on the practical operational needs of the electric ground-handling vehicle schedule, the integer programming model was established as follows:

$$\min f_1 = \sum_{i=1}^{n_v} \sum_{j=1}^{n_f} x_{ij} TD_{ij} \tag{1}$$

$$BT_{ij} + ST_{ij} \leq BT_{ij'}, \forall i \in V, \forall j, j' \in F \tag{2}$$

$$BT_{ij} + ST_{ij} \leq BT_{i'j}, \forall i, i' \in V, \forall j \in F \tag{3}$$

$$TD_{ij} = TT_{ij} \times S_i + C_i \times D_{jp}, \forall i \in V, \forall j \in F \tag{4}$$

The objective function (1) minimizes the total travel distance by all electric ground-handling vehicles. Constraint (2) ensures the right temporal logic when vehicle i serves two successive flights j and j'. Specifically, the starting serving for flight j' by vehicle i should not be earlier than the time at which vehicle i finishes the serving for flight j. Constraint (3) ensures the right temporal logic when vehicle i' serves flight j after vehicle i finishes the serving for flight j, where the priority of vehicle i' is lower than that of vehicle i. Constraint (4) represents the total distance traveled by electric ground-handling vehicle i during the serving of flight j, and in specific cases, it also includes the distance traveled to the depot p for charging.

$$VD_{ij} - D_{jp} \geq 0, \forall i \in V, \forall j \in F \tag{5}$$

$$VD_{ij} - TD_{ij'} \times S_i - D_{j'p} \leq 0, \forall i \in V, \forall j, j' \in F \tag{6}$$

$$CR_i = VD_i^m / CT_i^m, \forall i \in V \tag{7}$$

$$CT_i^c = C_i \times \left(VD_i^m - VD_{ij} - D_{jp}\right) / CR_i, \forall i \in V, \forall j \in F \tag{8}$$

$$FT_i = BT_{ij'} - BT_{ij} - ST_{ij} - C_i \times D_{jp}/S_i, \forall i \in V, \forall j, j' \in F \quad (9)$$

$$FT_i \geq CT_i^c, \forall i \in V \quad (10)$$

$$VD_{ij} \leq 0.2 \times VD_i^m, \forall i \in V, \forall j \in F \quad (11)$$

Constraints (5) and (6) represent the regular charging constraints. When both constraints are satisfied simultaneously, vehicle i must go to the depot p for charging. Constraint (5) indicates that when the vehicle i has just ended the serving by flight j, the electric ground-handling vehicle i must have sufficient battery level to travel the distance from its current location to the depot p for charging. Constraint (6) ensures that if vehicle i serves the next flight j', it must still have enough battery capacity to complete the service and return to the depot p for recharging. This constraint prevents the vehicle from getting stranded due to insufficient battery levels.

Constraint (7) represents the distance that electric ground-handling vehicle i can travel per unit of charging time. Constraint (8) represents the time required for the electric ground-handling vehicle i to charge from its current battery level to full capacity. Due to the potential for significant idle time between two successive flights, j and j', whose services are provided by the same vehicle i, it can be charged during the idle time. Constraint (9) represents the idle time of electric ground-handling vehicle i being occupied by two successive flights j and j'.

Constraints (10) and (11) describe the conditions that must be met to implement charging for the electric ground-handling vehicle i during the idle time. Specifically, the idle time FT_i for electric ground-handling vehicle i must first exceed the time CT_i^c required for its battery to charge from the current level to full capacity. Additionally, to minimize battery degradation and avoid charging the vehicle too frequently, charging is only considered when the vehicle's capacity is below 20%. When electric ground-handling vehicle i satisfies constraints (10) and (11), it will engage in idle time charging.

$$D_j \leq Q_i \quad (12)$$

$$x_{ij,}, C_i \in \{0, 1\}, \forall i \in V, \forall j \in F \quad (13)$$

$$n_{vs} \leq n_v \quad (14)$$

Constraint (12) guarantees that the demand for flight j does not exceed the capacity of vehicle i. Constraint (13) ensures that the binary variables x_{ij}, C_i can only take on values of 0 or 1. Constraint (14) guarantees that the number, n_{vs}, of occupied electric ground-handling vehicles should not exceed the total number, n_v, of that in the airport.

4. Proposed Scheduling Method

4.1. Construction Framework

The proposed method consists of two main components. First, the Mamba model serves as an encoder to process multi-dimensional features of flights and electric ground-handling vehicles. It combines position information, service requirements, and time window data through learnable embeddings, and uses its SSM-based architecture to capture complex inter-node relationships. Compared to traditional attention-based models, the Mamba model eliminates the quadratic complexity of self-attention, making it more computationally efficient, especially for large-scale scheduling problems. It effectively captures complex relationships between nodes for encoding, thereby generating feature representations that incorporate both spatial and temporal dependencies, providing strong support for subsequent decision-making.

Additionally, an attention-based neural network, functioning as a decoder, is designed to dynamically select the most suitable flights for electric ground-handling vehicles to service. By leveraging reinforcement learning, the decoder continuously improves decision-making efficiency, reducing reliance on manually designed heuristics and enabling better generalization across different scheduling scenarios. The decoder's parameters are optimized through reinforcement learning to generate efficient scheduling solutions. The training and solving processes of the algorithm are illustrated in Figure 4, demonstrating how the model learns to generate optimal scheduling solutions through the interaction between the Mamba model and the attention-based neural network.

Compared to conventional heuristic-based approaches, this framework not only enhances the quality of scheduling decisions but also significantly improves adaptability to dynamic airport environments. The combination of Mamba and attention mechanisms ensures a balance between solution accuracy and computational efficiency, making the algorithm particularly advantageous in real-time scheduling scenarios.

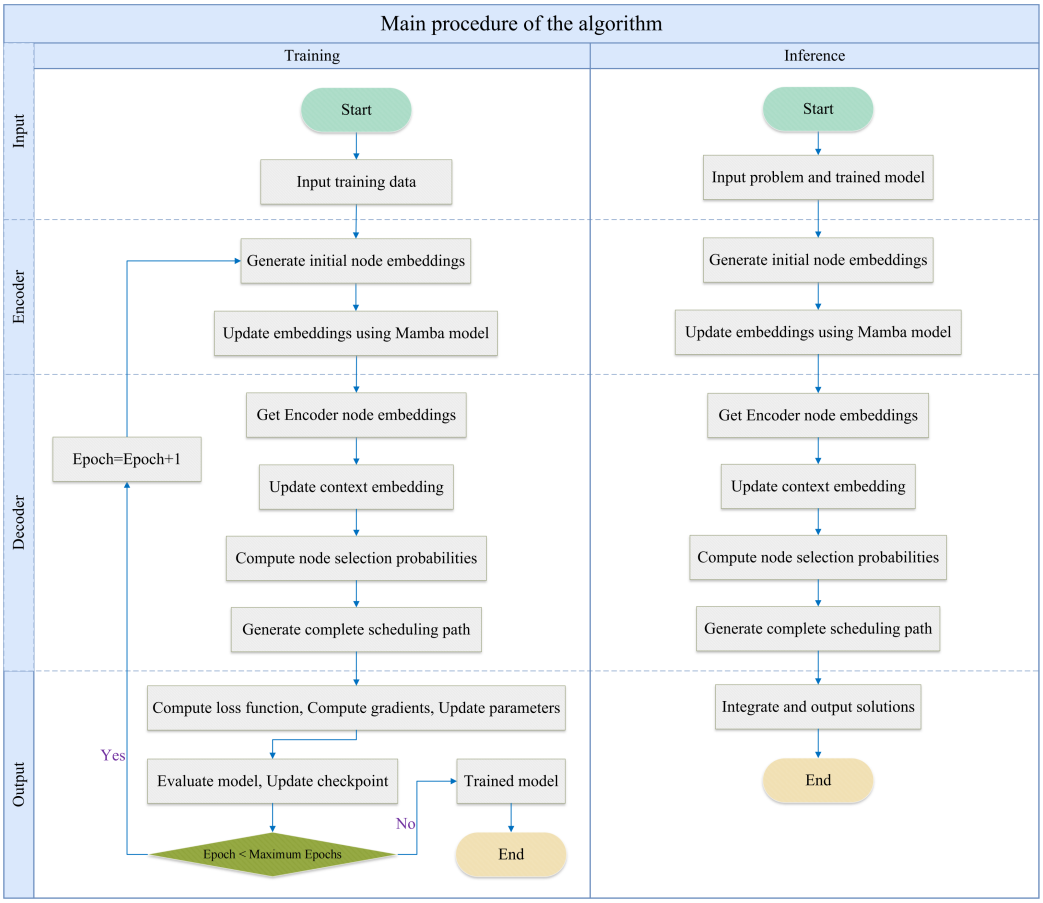


Figure 4. Main procedure of the algorithm training and solving.

4.2. MDP Formulation

The scheduling problem is approached by formulate the solution construction as a Markov decision process (MDP). At the t iteration, the agent systematically chooses the subsequent flight to service according to a learned strategy, culminating in a finite sequence of actions/episode, where:

1. State Space: The state space is defined as $S = \{s_i^t \mid 1 \leq t \leq T, i \in V\}$, where s_i^t represents the state of electric ground-handling vehicle i at time t. The state can include both static information, such as the vehicle's current location or the overall airport

layout (graph embeddings), and dynamic information, such as context embeddings (e.g., time of day, task load) and previously selected flights.

2. Probability Space: The probability space is denoted as $(\Omega, \mathcal{F}, P)$, where Ω is the sample space, $\mathcal{F}$ is the sigma-algebra, and P is the probability measure.
3. Markov Process: The process $\{s_i^t\}_{1 \le t \le T}$ is a Markov process on this probability space, where each state s_i^t is a random variable mapping from Ω to S, i.e., $s_i^t : \Omega \to S$. The transition probabilities depend on the current state, and the system evolves according to the Markov property, where the future state depends only on the present state and not on the past history.
4. Action Space: The action space is defined as $A = \{a_i^t \mid 1 \le t \le T, i \in V\}$. Each action $a_i^t \in A$ represents the decision made by vehicle i at time step t. Specifically, a_i^t represents the vehicle selecting the next flight or parking lot to go to from the node set $N = \{0, 1, \ldots, n_f\}$, where 0 indicates the depot. The actions need to satisfy constraints, including time windows, battery capacity, and vehicle accessibility.
5. Reward Function: After vehicle i completes all the flights to be served within T steps and obtains the complete solution $a_i = \{a_i^1, \ldots, a_i^T\}$, the total reward is represented as the negative length of the total distance, i.e., $R_{a_i} = -L(a_i)$. Since the objective is to minimize the total travel distance, maximizing the reward function is equivalent to minimizing the objective function $L(ai)$, thereby achieving optimal scheduling.
6. Transition Function: Define the deterministic state evolution $P_i = P(s_i^{t+1} | s_i^t, a_i^t)$ after each operation a_i^t, updating the vehicle state from state s_i^t to s_i^{t+1}.

4.3. Network Architecture

The proposed scheduling algorithm combines the Mamba model with an attention-based neural network to effectively solve the electric ground-handling vehicle scheduling problem. The architecture of the algorithm is illustrated in Figure 5. Given a scheduling instance d_i for vehicle i, which represents the input data describing the scheduling environment and constraints for that vehicle (including factors such as battery level, time windows, and flight assignments), the policy $\pi_\theta(a_i|d_i)$ is parameterized by θ and factorized as follows:

$$\pi_\theta(a_i|d_i) = \prod_{t=1}^{T} \pi_\theta(a_i^t | d_i, a_i^{1:t-1}), \forall i \in V \tag{15}$$

where $a_i^{1:t-1}$ represents the current partial solution (route) that has been constructed before the t^{th} iteration. At the t iteration, this study fix the previously constructed route $a_i^{1:t-1}$, and select an unvisited yet feasible flight (or depot) according to the policy π_θ, which will be added into the current route.

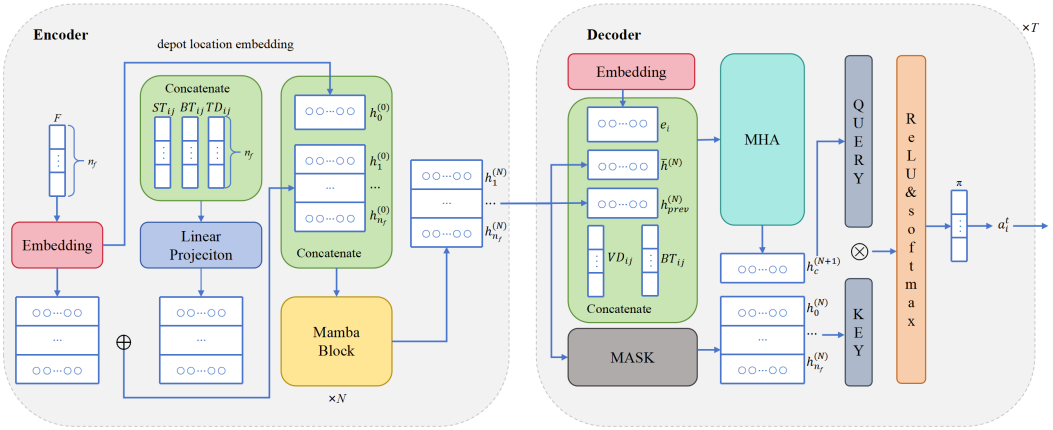


Figure 5. The architecture of the proposed scheduling algorithm.

4.4. Encoder

The encoder employs the Mamba model to comprehensively process multi-dimensional features of flights F and electric ground-handling vehicles V. Unlike traditional attention-based approaches, the Mamba model excels at capturing complex relationships between nodes through its state space modeling architecture [28]. This makes it particularly suitable for airport ground operations, where vehicles and flights interact with various spatial and temporal constraints. The encoding process consists of the following steps:

1. Construct initial features for each node by combining distance information D_{jp}, service requirements ST_{ij}, and time window constraints BT_{ij}.
2. Process these features through multiple Mamba blocks to capture node relationships and dependencies.
3. Generate final node representations through layer normalization and feature fusion.

The initial feature representation combines multiple information sources to create a comprehensive embedding of the scheduling scenario. For each node (flight in F or depot p), the model constructs an initial embedding that incorporates ST_{ij}, BT_{ij} and TD_{ij}. This multi-dimensional representation is formulated as:

$$h_j^{(0)} = \begin{cases} e_0, & j = 0 \\ e_j + W \cdot [ST_{ij}, BT_{ij}, TD_{ij}] + b, & j = 1, \ldots, n_f \end{cases}, \quad \forall i \in V \tag{16}$$

The matrix $W \in \mathbb{R}^{d \times 3}$ is a learnable weight matrix that projects the three-dimensional feature vector $[ST_{ij}, BT_{ij}, TD_{ij}]$ into the embedding space of dimension d. And the matrix W is shared across all vehicles, meaning it does not depend on i. However, since $ST_{ij}, BT_{ij}, TD_{ij}$ vary with both vehicle i and flight j, the resulting embedding $h_j^{(0)}$ is vehicle-dependent. The bias term $b \in \mathbb{R}^d$ is also learnable.

The initial features are processed through $N = 3$ layers of Mamba blocks to capture both local and global relationships between nodes. Each Mamba block employs state space modeling to effectively process node information, allowing the model to understand complex dependencies between different flights in F and vehicles in V. In each layer l, the feature update process consists of two key operations:

$$\hat{h}_j^{(l)} = \text{LayerNorm}^{(l)}\left(h_j^{(l-1)} + \text{MambaBlock}^{(l)}(h_j^{(l-1)}; h_0^{(l-1)}, \ldots, h_{n_f}^{(l-1)})\right), \forall j \in F \tag{17}$$

$$h_j^{(l)} = \text{LayerNorm}^{(l)}\left(\hat{h}_j^{(l)}\right), \forall j \in F \tag{18}$$

The layer normalization operations stabilize the training process and ensure effective gradient flow through the deep network structure. The residual connections ($h_j^{(l-1)}$ term) help maintain important information from earlier layers while allowing the model to learn additional features.

Through this hierarchical processing, the encoder produces the final representations $h_j^{(N)}$ that capture both spatial characteristics (based on D_{jp}) and temporal characteristics (based on ST_{ij}, BT_{ij}) as well as operational constraints. The feature representations enable the subsequent attention-based decoder to make effective scheduling decisions.

4.5. Decoder

The decoder utilizes an attention-based neural network to help electric ground-handling vehicles select appropriate flights to serve. This sequential decision-making process is crucial as each selection needs to consider not only the current service requirements but also the potential impact on future scheduling decisions. Based on the feature representations from the encoder, the decoder leverages a attention mechanism to compre-

hensively evaluate all possible vehicle-flight combinations and goes through the following steps to make scheduling decisions:

1. Integrate the encoder's feature representations with the current scheduling state to form the context embedding.
2. Update the context embedding through the attention mechanism, focusing on relevant features for the current decision.
3. Calculate selection scores for available flights based on the updated context embedding.
4. Generate a probability distribution for flight selection to determine which flight should be served next.

This sequential decision-making process enables the model to generate scheduling solutions that effectively balance service efficiency and operational constraints.

Firstly, the decoder generates a context embedding that represents the dynamic scheduling state at each decision point. At each decision point for electric ground-handling vehicle i and flight j, this embedding combines vehicle status and scheduling history to provide a comprehensive view of the decision-making context:

$$h_c^{(N)} = [\overline{h}^{(N)}, e_i, h_{prev}^{(N)}, VD_{ij}, BT_{ij}], \forall i \in V, \forall j \in F \tag{19}$$

where $\overline{h}^{(N)}$ represents the uniform pooling of embeddings from the Mamba encoder; e_i represents the current vehicle i, used to distinguish it from other vehicles; $h_{prev}^{(N)}$ is the embedding of the previously served flight; VD_{ij} represents the remaining driving distance capacity of vehicle i; BT_{ij} represents the current service start time for vehicle i.

The state update mechanism is crucial for maintaining accurate representations of vehicle conditions and temporal constraints throughout the scheduling process. After each scheduling decision, the system updates two key state variables: the cruising range and the service start time. The cruising range update considers whether the vehicle has returned to the depot for charging or has consumed distance serving a flight. Similarly, the temporal state update accounts for charging time at the depot or the combination of travel time and service duration:

$$VD_{i,j'} = \begin{cases} VD_i^m, & \text{if } C_i = 1 \\ VD_{ij} - TD_{ij}, & \text{otherwise} \end{cases}, \quad \forall i \in V, \forall j, j' \in F \tag{20}$$

$$BT_{i,j'} = BT_{ij} + ST_{ij} + C_i \times CT_i^c + TT_{ij'}, \forall i \in V, \forall j, j' \in F \tag{21}$$

where TD_{ij} represents the travel distance for serving flight j, CT_i^c is the charging time needed for vehicle i to reach full capacity, TT_{ij}^s denotes the travel time during service, ST_{ij} is the service time that vehicle i is occupied by flight j, $BT_{ij'}$ represents the start time of the next potential flight j', and $TT_{ij'}$ is the travel time for electric ground-handling vehicle i serving for flight j'.

The decision-making process utilizes an attention mechanism to effectively evaluate the relationships between the current vehicle state and potential service flights. The context embedding first undergoes an multi-head attention (MHA) (M = 8 heads) layer to assess all possible service decisions:

$$h_c^{(N+1)} = \text{MHA}(h_c^{(N)}, H^{(N)}) \tag{22}$$

where $H^{(N)}$ contains the feature representations of all nodes from the Mamba encoder. Based on this updated context, the model calculates selection scores for each feasible flight using a combination of attention mechanisms and non-linear transformations:

$$v_t(j) = \begin{cases} \text{ReLU}\left(\frac{(W_Q h_c^{(N+1)})^T (W_K h_j^{(N)})}{\sqrt{d}} + b\right), & \text{if } j \text{ is feasible} \\ -\infty, & \text{otherwise} \end{cases}, \quad \forall j \in F \tag{23}$$

The feasibility of each flight is determined by comprehensive checks of operational constraints, including remaining travel distance capacity (VD_{ij}), service time windows (BT_{ij}), and whether the flight has already been served.

The matrices $W_Q \in \mathbb{R}^{d \times d}$ and $W_K \in \mathbb{R}^{d \times d}$ are learnable weight matrices used to transform the embedding vectors into query and key representations, respectively. These transformations are essential for computing the attention scores, which determine the selection probability of feasible flights.

The final stage of the decoding process involves converting the calculated scores into a probability distribution over possible flight selections. This is achieved through the Softmax function:

$$\pi_\theta(a_i|d_i, a_i^{1:t-1}) = \text{Softmax}(v_t(a_i)), \forall i \in V, \forall j \in F \tag{24}$$

The complete scheduling policy is then formed by combining the sequential decisions into a joint probability distribution:

$$\pi_\theta(a_i|d_i) = \prod_{t=1}^{T} \pi_\theta(a_i|d_i, a_i^{1:t-1}), \forall i \in V, \forall j \in F \tag{25}$$

This probabilistic approach allows the model to learn optimal scheduling strategies through reinforcement learning while maintaining the flexibility to adapt to different scenarios.

4.6. Policy Training

Given the policy parameterized by the neural network π_θ in Section 4.5, this study trains the policy network by exploiting the REINFORCE algorithm [29] with a rollout baseline [30]. The baseline helps reduce the variance of gradient estimates while maintaining the unbiased nature of the REINFORCE algorithm, which is particularly important given the complex constraints in electric ground-handling vehicle scheduling.

For each vehicle $i \in V$, the objective is to minimize the total travel distance:

$$L(\theta|d_i) = \mathbb{E}_{\pi_\theta(a_i|d_i)}[L(a_i)], \forall i \in V \tag{26}$$

where, the objective function minimizes the total travel distance across all vehicles rather than optimizing each vehicle's distance independently. The function $L(a_i)$ represents the total travel distance of vehicle i, which is computed as the sum of the distances traveled between consecutive locations in the assigned route.

The expectation operator $\mathbb{E}_{\pi_\theta(a_i|d_i)}$ represents the expected value under the learned policy π_θ, ensuring that the optimization accounts for the stochastic nature of scheduling decisions.

This formulation explicitly accounts for the distance-related operational costs while implicitly considering the battery capacity and charging time constraints through the policy network's decision-making process.

The gradient update formula used in the training process is:

$$\nabla L(\theta|d) = \sum_{z \in \{z_b\}_{b=1}^{B}} \frac{1}{n_v} \sum_{i=1}^{n_v} [(L(a_i) - BS(d_i)) \nabla \log \pi_\theta(a_i|d_i)], \forall i \in V \tag{27}$$

where z_b represents the random instance b in batch, B represents the batch size and $BS(d_i)$ represents the baseline solution cost for vehicle i, calculated using the best checkpoint model. This rolling baseline approach helps stabilize training by providing a dynamic reference point for policy improvement. The network parameters are then updated using the Adam optimizer [31].

To ensure the reliability of training and prevent overfitting, this study employs a multi-step validation mechanism. After each epoch, this study evaluates the trained model on a separate validation set X_{val}, distinct from the training data, to measure generalization performance. The evaluation involves computing the objective function values on validation instances and comparing them against previous best solutions. The comparison employs a paired t-test [32] to determine if the current model achieves statistically significant improvements over the best checkpoint. This statistical approach helps filter out random fluctuations in performance and ensures that model updates are based on genuine improvements in scheduling capability.

In this context, we define a random variable D_i representing the difference in performance metrics between the current model and the best checkpoint for each validation sample i in X_{val}. The mean difference $\bar{D}$ and standard deviation s_D are used as estimators of the parameters of this random variable. We conduct the paired t-test under the assumption that the differences are normally distributed and independent, which are standard assumptions for this type of statistical test. The significance level α is set at 5%.

Additionally, an early stopping criterion is applied to prevent unnecessary training iterations. If the model does not show statistically significant improvement over several consecutive epochs, the training is terminated to avoid overfitting and computational waste. When significant improvement is detected, the checkpoint is updated and a new validation set is generated to maintain the robustness of the evaluation process. This dynamic validation strategy ensures that the trained model maintains a balance between exploration (improving scheduling performance) and stability (avoiding over-optimization on specific training instances). The complete training process is detailed in Algorithm 1.

In summary, this training framework optimizes the policy network to learn efficient scheduling strategies while maintaining solution feasibility through explicit consideration of operational constraints. The combination of policy gradient learning with a rolling baseline, multi-step validation, and early stopping criteria creates a robust training process that can effectively handle the complexities of electric ground-handling vehicle scheduling.

5. Experimental Results and Analysis

This study evaluates the performance of the algorithm using airport electric ground-handling vehicle scheduling cases. Specifically, statistical data from Xiamen Gaoqi Airport is utilized, as well as 10 different electric ground-handling vehicles with 4 priority levels, as shown in Figures 1 and 2. The vehicles used and the services they provide are shown in Figures 1 and 2. Xiamen Gaoqi Airport is selected as the study subject. The airport includes 2 terminals and 89 parking stands. To conduct the experiment, we loaded the airport map, which includes information such as the locations of Terminals, gates, and parking stands [33] into SUMO, and automatically calculated the distances using SUMO. Based on flight data from Xiamen Gaoqi Airport [34], experimental cases containing 20, 50, 100, and 200 flights are created. Detailed flight information can be found in Tables 2 and 3. To comprehensively evaluate the algorithm's performance and validate its effectiveness, various scheduling cases were designed and compared against the CPLEX solver, recent heuristic algorithms, and self-constructed heuristic algorithms. Additionally, cases involving real-time flight arrivals are simulated to test the algorithm's capability to handle dynamic flight changes.

Algorithm 1 Policy Optimization for Electric Ground-handling Vehicle Scheduling

Input: epochs E; iterations per epoch T; number of vehicles n_v; batch size B; significance level α
Initialize parameters θ, best checkpoint $\theta^* \leftarrow \theta$
for $epoch = 1$ to E **do**
  for $iteration = 1$ to T **do**
    Generate random samples $\{z_b\}_{b=1}^{B}$
    $\{\{B_i^b\}_{b=1}^{B}\}_{i=1}^{n_v} \leftarrow \text{Greedy}(\{z_b\}_{b=1}^{B}, \theta^*)$
    for $i = 1$ to n_v **do**
      $\{L_i^b\}_{b=1}^{B} \leftarrow \text{Sample}(\{d_i^b\}_{b=1}^{B}, \theta)$
      Update $VD_{ij'}$ and $BT_{ij'}$ according to equations (20) and (21)
    end for
    Compute gradient $\frac{1}{B}\nabla L$ using equation (27)
    Update parameters $\theta \leftarrow \text{Adam}(\theta, \frac{1}{B}\nabla L)$
  end for
  $\{\{L_i^b\}_{b=1}^{|X_{\text{va}}|}\}_{i=1}^{n_v} \leftarrow \text{Greedy}(X_{\text{va}}, \theta)$
  $\{\{B_i^b\}_{b=1}^{|X_{\text{va}}|}\}_{i=1}^{n_v} \leftarrow \text{Greedy}(X_{\text{va}}, \theta^*)$
  if t-test$(\sum_{i=1}^{n_v}\{L_i^b\}_{b=1}^{|X_{\text{va}}|}, \sum_{i=1}^{n_v}\{B_i^b\}_{b=1}^{|X_{\text{va}}|}) < \alpha$ **then**
    Update best checkpoint $\theta^* \leftarrow \theta$
    Generate new validation set X_{va}
  end if
end for

Table 2. Airport arrival data.

Flight Number	Arrival Time	Flight Number	Arrival Time
SC4786	06:30	SC2269	06:50
MF8093	06:35	MF8275	06:50
MF8093	06:35	MF8275	06:55
MF8093	06:40	MF8275	06:55
SC2269	06:45	SC2135	06:55
...	...	...	...

Table 3. Airport departure data.

Flight Number	Departure Time	Flight Number	Departure Time
BK2765	08:25	HU7047	08:50
CZ3723	08:30	MU2939	08:50
FM9263	08:40	9C8815	08:55
HO1105	08:45	SC4781	09:40
ZH8651	08:45	9C8975	09:50
...	...	...	...

The instance generation process is inspired by the experimental setup for ground-handling vehicle instances in [35], and adapted to fit the operational characteristics of the studied scenario. For each vehicle, flight demands are uniformly sampled from [1, ..., 12]. Then, for 20, 50, 100, and 200 flights, the vehicle capacities are set to 30, 40, 50, and 60, respectively. Based on the specifications of electric ground-handling vehicles in real-world airport settings and research on vehicle performance characteristics by Sanz [36], the vehicle speed is set to 30 km/h, and the cruising range is set to 80 km. The policy network is trained for 200 epochs. To match reality, 12,800 instances are generated for each epoch based on flight information from Xiamen Gaoqi Airport. To validate model performance, we follow the approach of Kool et al. and randomly generate 1000 cases with the same scale and random seed at the end of each epoch as the validation set to evaluate the model performance. In the test phase, 1000 cases with 20, 50, 100, and 200 flights are generated

using different random seeds. Finally, during testing, comparisons are made with CPLEX, several representative heuristic algorithms, and constructed heuristic algorithms. The following algorithms are used for comparison:

1. CPLEX, a powerful optimization solver used for solving complex linear, integer, and quadratic programming problems efficiently [37].
2. Iterated Local Search (ILS) is a heuristic search algorithm that escapes local optima by perturbing solutions obtained from local searches and then re-applying local search to find the global optimum [38].
3. Ant Colony System-Simulated Annealing (ACS-SA) is a hybrid optimization algorithm that combines the collective intelligence of Ant Colony Systems with the stochastic search capabilities of Simulated Annealing to solve optimization problems [39].
4. The LNS-SA method, combining LNS [10] and SA [6], utilizes the Metropolis criterion for solution acceptance to ensure quality, offering an efficient algorithm for optimization by iteratively improving existing solutions through local search and leveraging the benefits of SA's annealing process [40].
5. Genetic Algorithm-Ant Colony Optimization (GA-ACO) is a hybrid heuristic optimization strategy that combines the global search capabilities of Genetic Algorithms with the local search capabilities of Ant Colony Optimization to provide higher quality and more stable solutions [41].
6. Constraint Programming-Large Neighborhood Search (CP-LNS) is an algorithm that combines CP with LNS, using CP techniques to guide the search process and employing LNS strategies to explore the solution space and find optimal solutions [42].
7. Insertions, a type of construction heuristics to solve VRPs. The comparison is made with random, nearest, farthest insertion, and nearest neighbor [43].

This study compared the average performance of different methods in terms of objective values and relative gaps, where the gaps were calculated based on the best solution found across all methods. The decoding framework consists of two strategies: (1) Greedy strategy, where at each time step the algorithm select the flight with maximum probability based on the policy network's predictions; (2) Sampling strategy, where first generates a diverse set of 1000 candidate solutions by sampling from the policy network's probability distributions, then picks the solution with the best performance as output. Additionally, this study recorded the average processing time per case for each the algorithm, which varies depending on the specific approach used. The experiments were run on a machine equipped with a GeForce RTX-4060 GPU and an Intel i7-12650H CPU running at 4.70 GHz.

For CPLEX and heuristic algorithms, a time limit of 60 min is set for solving cases of sizes 20, 50, 100, and 200. For the heuristic algorithms, the population size is uniformly set to 200, the crossover probability is set to 0.8, and the mutation probability is set to 0.3. For the algorithm designed in this study, the policy network is trained for 200 epochs using the flight data from Xiamen Gaoqi Airport. Each training epoch consists of 200 batches, with a batch size of 64 instances, and at the end of each epoch, 1000 instances are evaluated. In the neural structure, the Mamba model uses 3 layers in the encoder, and each multi-head attention layer in the decoder uses 8 heads. The embedding dimension of the algorithm is set to 128. The Adam optimizer is used with a constant learning rate of 10^{-4} to update parameters. To validate the performance of the algorithm, the significance level of the t-test is set to 5%.

5.1. Performance Comparison of the Proposed Algorithm

The objective function values, gaps, and solution times of all compared algorithms in different cases are shown in Table 4. The objective function values and gap results obtained by the algorithms are shown in Figure 6, while the heatmap of computational

times is shown in Figure 7. From Figure 6 and Table 4, it is clear that the algorithm designed in this study consistently achieves the lowest cost across all cases. Particularly, when using the sampling strategy for decoding, the average gap approaches 0. Furthermore, the greedy strategy designed in this study also results in lower costs and gaps compared to other algorithms.

For the case with 20 flights, the ACS-SA algorithm, the best among the other comparison algorithms, generates the shortest total distance. Compared to ACS-SA, the sampling strategy in this study reduces the cost by 9.19%, while the greedy strategy achieves a 4.18% reduction.

When considering 50 flights, the LNS-SA algorithm leads with the lowest cost among the other comparison algorithms. In this case, the sampling strategy reduces the cost by 16.76% compared to LNS-SA, and the greedy strategy achieves a 11.32% reduction. With 100 flights, the CP-LNS algorithm produces the best scheduling solution among the other comparison algorithms. Here, the sampling strategy outperforms CP-LNS, reducing the cost by 20.92%, and the greedy strategy reduces the cost by 16.28%.

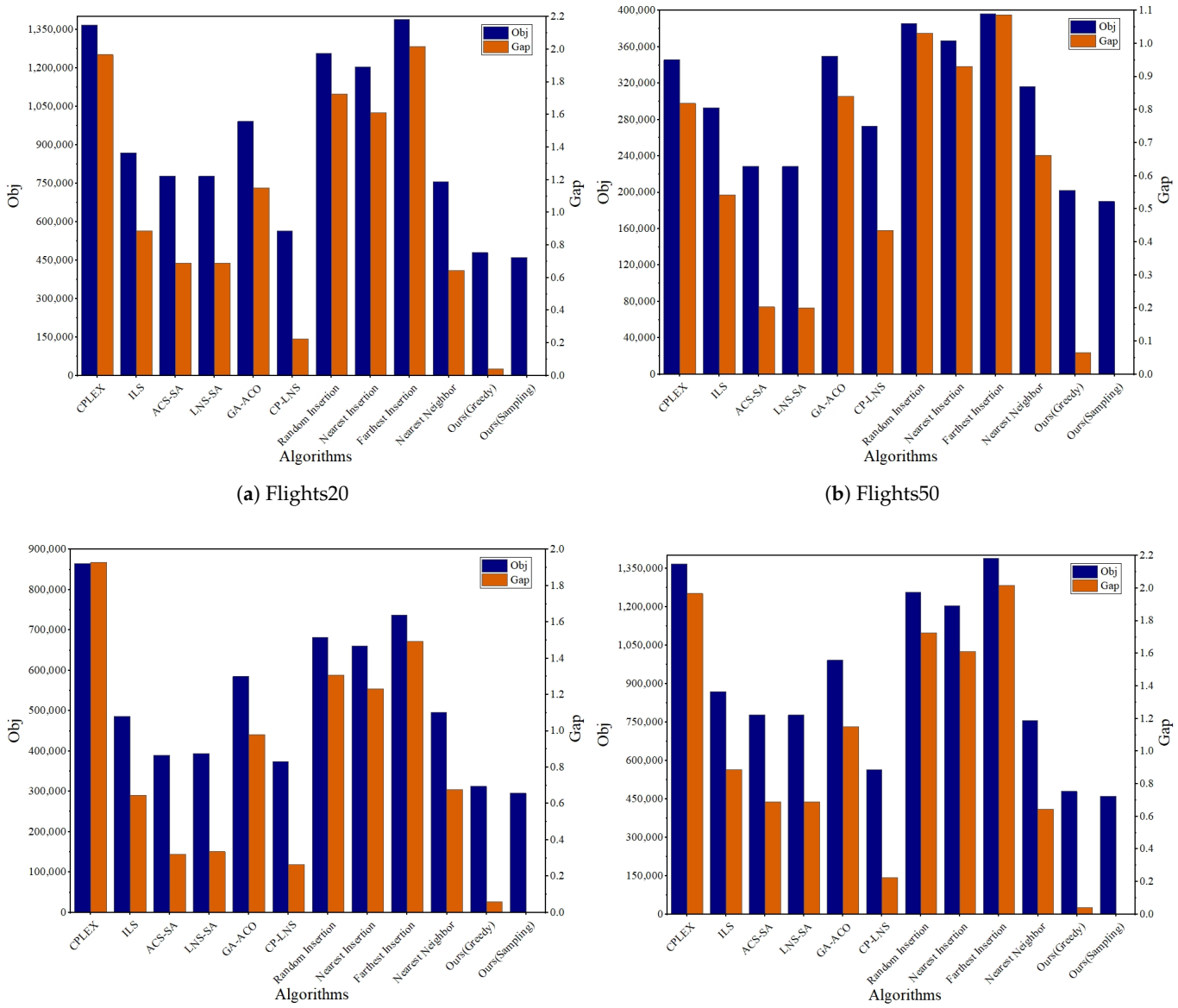
Finally, for 200 flights, CP-LNS again provides the best solution among the other comparison algorithms. The sampling strategy reduces the cost by 28.50% compared to CP-LNS, and the greedy strategy reduces it by 22.39%. These comparison results strongly demonstrate the superior performance of the algorithm designed in this study for the electric ground-handling scheduling problem.

The solution time of the algorithms is further analyzed in Figure 7, which shows that the algorithm developed in this study has the fastest computation time, outperforming all others except for the Nearest Neighbor algorithm. However, from Table 4 and Figure 6, it can be observed that the objective function values obtained by the algorithm designed in this study are much lower than those obtained by the Nearest Neighbor algorithm. Even with a larger number of flights, the computation time difference between the algorithm developed in this study and the Nearest Neighbor algorithm remains small, and is still much less than the time required by other algorithms, which can take several minutes or even an hour to complete. This is because the algorithm designed in this study directly constructs the solution for the case without the need for additional iterative improvement steps, making it faster than heuristic methods. Therefore, it can be concluded that the algorithm designed in this study produces the optimal solution in a reasonable time, proving its superiority over other best algorithms.

In conclusion, the experiments fully demonstrate the superior performance of the algorithm designed in this study for solving the electric ground-handling vehicle scheduling problem. This algorithm can quickly generate scheduling solutions with significant cost benefits, which is crucial for improving airport operational efficiency, reducing operational costs, and minimizing environmental pollution.

Table 4. Comparison with different algorithms in terms of Obj (total travel distance), Gap (algorithmic performance deviation) and computation time.

Algorithm	Flights20			Flights50			Flights100			Flights200		
	Obj	Gap	Time	Obj	Gap	Time	Obj	Gap	Time	Obj	Gap	Time
CPLEX [37]	157,233.98	51.25%	60 m	345,766.61	81.92%	60 m	865,651.23	192.86%	60 m	1,367,551.07	196.71%	60 m
ILS [38]	129,823.91	24.88%	60 m	293,126.97	54.22%	60 m	486,324.92	64.53%	60 m	869,550.54	88.66%	60 m
ACS-SA [39]	114,484.43	10.13%	60 m	228,734.39	20.34%	60 m	390,291.67	32.04%	60 m	779,122.64	69.04%	60 m
LNS-SA [40]	117,582.26	13.11%	60 m	228,354.01	20.14%	60 m	394,664.15	33.52%	60 m	778,222.18	68.84%	60 m
GA-ACO [41]	161,237.22	55.10%	60 m	349,830.54	84.05%	60 m	585,206.93	97.98%	60 m	991,725.81	115.17%	60 m
CP-LNS [42]	132,149.59	27.12%	60 m	272,574.76	43.41%	60 m	373,779.32	26.45%	60 m	564,015.47	22.37%	60 m
Random Insertion [43]	186,222.47	79.13%	2.58 s	385,925.72	103.04%	0.51 m	682,444.78	130.88%	2.57 m	1,256,199.28	172.55%	0.46 h
Nearest Insertion [43]	173,834.83	67.22%	3.21 s	367,056.41	93.12%	0.61 m	660,099.82	123.32%	3.37 m	1,203,523.53	161.12%	0.60 h
Farthest Insertion [43]	175,092.68	68.43%	3.53 s	396,680.03	108.70%	0.62 m	736,720.21	149.24%	4.18 m	1,390,207.03	201.63%	0.64 h
Nearest Neighbor [43]	164,672.94	58.40%	0.99 s	315,982.31	66.24%	1.13 s	495,522.39	67.64%	1.39 s	757,150.84	64.27%	1.85 s
Ours (Greedy)	109,694.07	5.52%	3.76 s	202,505.72	6.54%	4.23 s	312,894.87	5.85%	4.96 s	480,325.72	4.21%	7.66 s
Ours (Sampling)	103,953.90	0.00%	3.86 s	190,064.83	0.00%	4.56 s	295,581.62	0.00%	6.01 s	460,898.34	0.00%	10.77 s



(**a**) Flights20 (**b**) Flights50

(**c**) Flights100 (**d**) Flights200

Figure 6. Comparison of Obj (total travel distance) and Gap (algorithmic performance deviation) among algorithms across flight numbers.

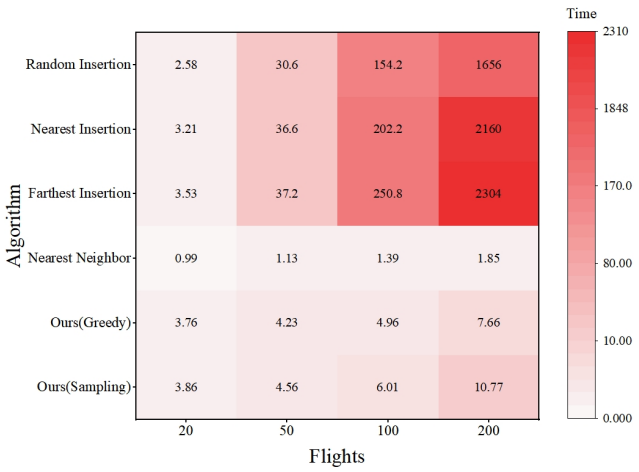


Figure 7. Computational time analysis heatmap for different algorithms across flight numbers.

5.2. Generalization Ability of the Proposed Algorithm

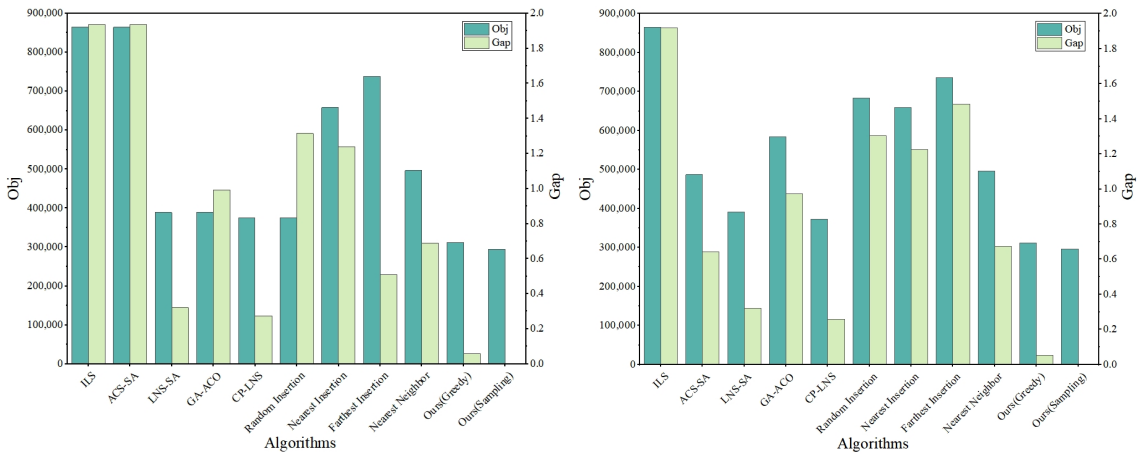
To verify the algorithm's ability to maintain good performance in diverse, unseen scenarios, this study evaluates the model's generalization ability by using cases sampled from different distributions. Specifically, to simulate fluctuations in flight demand across various scenarios, flight demand was sampled from a Gaussian distribution with a mean of 6 and variance of 3, and from a Poisson distribution with an expected event count of 6. Additionally, the arrival times of the flights were sampled based on the hourly distribution probability of Xiamen Gaoqi Airport, which follows a Gaussian distribution with a mean of 0, variance of 1, and a Poisson distribution with an expected event count of 5.

The objective function values and gap results obtained by the algorithms are shown in Figure 8, and the computational times are shown in Figure 9. From Figures 8 and 9, it is evident that our algorithm achieves the lowest objective value and gap in a shorter time. Further analysis of the data in Table 5 reveals that, in the Gaussian distribution case, the CP-LNS algorithm generates the shortest total distance among the other comparison algorithms. Compared to CP-LNS, the sampling strategy in this study reduces the cost by 27.49%, while the greedy strategy reduces the cost by 20.41%.

Similarly, in the Poisson distribution case, the CP-LNS algorithm again produces the shortest total distance among the other comparison algorithms. Here, the sampling strategy reduces the cost by 16.32% compared to CP-LNS, and the greedy strategy reduces it by 20.56%. These experiments demonstrate that the algorithm designed in this study has good generalization ability for both of the aforementioned distributions.

Table 5. Generalization study on varying parameters for electric ground-handling scheduling, focusing on Obj (total travel distance) and Gap (algorithmic performance deviation) and computation time.

Algorithm	Gaussian			Poisson		
	Obj	Gap	Time	Obj	Gap	Time
ILS [38]	864,298.73	193.75%	60 m	864,907.54	191.87%	60 m
ACS-SA [39]	486,324.92	65.28%	60 m	487,068.61	64.36%	60 m
LNS-SA [40]	388,939.14	32.18%	60 m	391,035.36	31.96%	60 m
GA-ACO [41]	586,559.46	99.35%	60 m	584,463.24	97.23%	60 m
CP-LNS [42]	375,131.85	27.49%	60 m	373,035.63	25.88%	60 m
Random Insertion [43]	681,092.25	131.48%	2.83 m	683,188.47	130.55%	3.76 m
Nearest Insertion [43]	658,747.29	123.89%	3.64 m	659,356.13	122.51%	4.81 m
Farthest Insertion [43]	738,072.74	150.84%	5.98 m	735,976.52	148.36%	6.46 m
Nearest Neighbor [43]	496,874.92	68.87%	1.42 s	496,266.08	67.47%	1.68 s
Ours (Greedy)	311,542.34	5.88%	3.76 s	312,151.18	5.34%	5.13 s
Ours (Sampling)	294,229.09	0.00%	6.32 s	296,325.31	0.00%	6.25 s



(**a**) Gaussian (**b**) Poisson

Figure 8. Comparison of Obj (total travel distance) and Gap (algorithmic performance deviation) among algorithms across distributions.

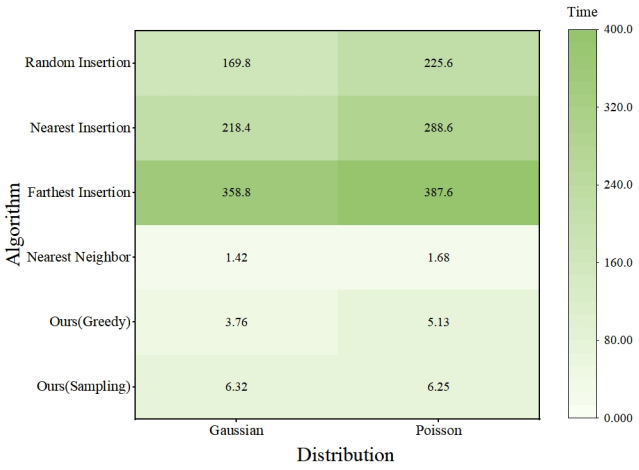


Figure 9. Computational time analysis heatmap for different algorithms across distributions.

5.3. Real-Time Scheduling Performance of the Proposed Algorithm

To validate the algorithm's ability to handle real-world scenarios, its performance in processing randomly arriving real-time flights was tested. To evaluate this ability, the algorithm designed in this study was compared with a fast heuristic algorithm. First, algorithms were adjusted to operate in a re-optimization mode to simulate the real-time dynamics of an airport. When a new flight arrives, the algorithm needs to determine how to arrange the electric ground-handling vehicles to service the flight within the flight's time window. In this study, 1000 cases containing 20 flights were generated for testing, and these cases were gradually expanded to 200 flights during inference.

Further analysis of the data in Table 6 reveals that, among the compared algorithms for real-time scheduling problems, the Nearest Neighbor algorithm achieved the shortest computational time (4.13 s). This indicates that Nearest Neighbor can provide a quick scheduling decision, which is beneficial in highly dynamic airport environments where rapid response times are required. However, this efficiency comes at the cost of higher total scheduling costs, as the algorithm does not consider global optimization.

Table 6. Results of real-time electric ground-handling scheduling considering Obj (total travel distance) and Gap (algorithmic performance deviation) and computation time.

Algorithm	**Flights20 → Flights200**		
	Obj	**Gap**	**Time**
Random Insertion [43]	1,307,433.28	155.29%	2.90 m
Nearest Insertion [43]	1,254,757.53	145.01%	3.86 m
Farthest Insertion [43]	1,441,441.03	181.46%	3.74 m
Nearest Neighbor [43]	808,384.84	57.85%	4.13 s
Ours (Greedy)	531,559.72	3.79%	11.87 s
Ours (Sampling)	512,132.34	0.00%	15.78 s

In contrast, the algorithm proposed in this study significantly improves scheduling performance. These reduction percentages are calculated based on the objective values presented in Table 6, by comparing the total costs of different strategies. The results show that the sampling strategy in this study significantly reduces the cost by 36.64%, while the greedy strategy reduces the cost by 34.24%. This comparison highlights the superior optimization capability of the proposed algorithm in real-time scheduling scenarios.

Moreover, the results in Table 6 demonstrate that the algorithm developed in this study not only achieves competitive solution quality but also maintains a reasonable computational time (11.87 s for the greedy strategy and 15.78 s for the sampling strategy). Although it requires slightly more time than the Nearest Neighbor algorithm, it provides substantially

lower costs, making it a more effective choice for real-world airport operations. The ability to generate near-optimal scheduling decisions within seconds ensures that the proposed algorithm can be effectively deployed in dynamic, high-demand airport environments.

Overall, the proposed algorithm successfully balances solution quality and computational efficiency. By leveraging advanced techniques such as the Mamba model and an attention-based neural network, it not only enhances adaptability but also ensures real-time decision-making capabilities. These advantages make the algorithm well-suited for electric ground-handling vehicle scheduling in large-scale airport operations.

Overall, the algorithm proposed in this study demonstrates excellent performance in electric ground-handling vehicle scheduling problems. Especially with the sampling strategy adopted by the algorithm, it successfully balances solution quality and computation speed, making rapid decisions without sacrificing optimality. This characteristic makes the algorithm particularly suitable for dynamic, high-demand airport environments. By incorporating advanced techniques such as the Mamba model and attention-based neural network, the algorithm also shows outstanding adaptability, enabling it to efficiently generate real-time solutions. The comparative experiments presented above fully demonstrate the wide application potential and immense value of this algorithm in airport and other fields.

6. Conclusions

This paper conducts an in-depth investigation into the scheduling problem of electric ground-handling vehicles, presenting a novel algorithm for scheduling electric ground-handling vehicles, combining the MAMBA model with an attention-based neural network. The MAMBA model processes multi-dimensional features, capturing complex relationships between scheduling elements such as flight information, vehicle locations, and time window constraints. The attention-based network dynamically incorporates data to select the most suitable flights for electric ground-handling vehicles. The algorithm is validated using operational data from Xiamen Gaoqi International Airport. Experimental results demonstrate that the proposed algorithm outperforms classical metaheuristics, construction heuristics, and the CPLEX solver, effectively solving the scheduling problem of assigning electric ground-handling vehicles to numerous flights and delivering high-quality solutions. It also demonstrates strong generalization capabilities, handling scheduling problems with varying parameters and real-time scheduling scenarios. In conclusion, the proposed algorithm exhibits excellent performance in solving the scheduling problem of electric ground-handling vehicles considering charging, providing airlines with optimal scheduling solutions for electric ground-handling vehicles. Future research will focus on the following directions: (1) applying the proposed method to larger scheduling instances with more flights to evaluate its scalability; (2) integrating heterogeneous electric ground-handling vehicles with different battery capacities and service constraints to enhance the model's practical applicability; (3) further exploring how to incorporate energy consumption during the service process into the scheduling scheme to develop a more comprehensive and precise optimization strategy; (4) considering a broader range of aircraft types and ground-handling vehicles to better account for the diverse operational requirements across different aircraft models.

Author Contributions: Conceptualization, J.L. and W.F.; methodology, J.L., W.F., K.L. and J.Z.; validation, J.L., W.F., Y.F. and G.H.; writing—original draft preparation, J.L. and W.F.; writing—review and editing, J.L. and W.F.; supervision, J.L., W.F., G.H., Y.F., K.L. and J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Civil Aviation Education and Talent Project of China, grant number MHJY2024038.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhao, P.X.; Gao, W.Q.; Han, X.; Luo, W.H. Bi-objective collaborative scheduling optimization of airport ferry vehicle and tractor. *Int. J. Simul. Model.* **2019**, *18*, 355–365. [CrossRef]
2. Wang, Y.; Sun, J.; Sun, B.; Feng, M. Research on Application of Airport Tanker Truck Scheduling Based on Particle Swarm Optimization. In Proceedings of the 2021 6th International Conference on Control, Robotics and Cybernetics (CRC), Shanghai, China, 9–11 October 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 278–282.
3. Han, X.; Zhao, P.; Meng, Q.; Yin, S.; Wan, D. Optimal scheduling of airport ferry vehicles based on capacity network. *Ann. Oper. Res.* **2020**, *295*, 163–182. [CrossRef]
4. Guo, W.; Xu, P.; Zhao, Z.; Wang, L.; Zhu, L.; Wu, Q. Scheduling for airport baggage transport vehicles based on diversity enhancement genetic algorithm. *Nat. Comput.* **2020**, *19*, 663–672. [CrossRef]
5. Feng, X.; Zuo, H.; Sun, Q. Research on collaborative scheduling of aircraft ground service vehicles based on simple temporal network. In Proceedings of the 2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT), Changsha, China, 20–22 October 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 263–269.
6. Zampirolli, K.A.; Amaral, A.R.S. Simulated annealing and iterated local search approaches to the aircraft refueling problem. In *Computational Science and Its Applications—ICCSA 2021: 21st International Conference, Cagliari, Italy, 13–16 September 2021;* Proceedings, Part IV 21; Springer International Publishing: Berlin/Heidelberg, Germany, 2021; pp. 422–438.
7. Han, X.; Zhao, P.; Kong, D. Two-Stage Optimization of Airport Ferry Service Delay Considering Flight Uncertainty. *Eur. J. Oper. Res.* **2023**, *307*, 1103–1116. [CrossRef]
8. Liu, Y.; Wu, J.; Tang, J.; Wang, W.; Wang, X. Scheduling optimisation of multi-type special vehicles in an airport. *Transp. B Transp. Dyn.* **2022**, *10*, 954–970. [CrossRef]
9. Zhang, J.; Chong, X.; Wei, Y.; Bi, Z.; Yu, Q. Optimization of apron support vehicle operation scheduling based on multi-layer coding genetic algorithm. *Appl. Sci.* **2022**, *12*, 5279. [CrossRef]
10. Padrón, S.; Guimarans, D. An improved method for scheduling aircraft ground handling operations from a global perspective. *Asia-Pac. J. Oper. Res.* **2019**, *36*, 1950020. [CrossRef]
11. Zhou, J.; Wu, Y.; Cao, Z.; Song, W.; Zhang, J.; Chen, Z. Learning Large Neighborhood Search for Vehicle Routing in Airport Ground Handling. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 9769–9782. [CrossRef]
12. Zhao, B.; Gui, H.; Li, H.; Xue, J. Cold chain logistics path optimization via improved multi-objective ant colony algorithm. *IEEE Access* **2020**, *8*, 142977–142995. [CrossRef]
13. Attar, S.F.; Mohammadi, M.; Pasandideh, S.H.R.; Naderi, B. Formulation and exact algorithms for electric vehicle production routing problem. *Expert Syst. Appl.* **2022**, *204*, 117292. [CrossRef]
14. Breunig, U.; Baldacci, R.; Hartl, R.F.; Vidal, T. The electric two-echelon vehicle routing problem. *Comput. Oper. Res.* **2019**, *103*, 198–210. [CrossRef]
15. Ganji, M.; Kazemipoor, H.; Molana, S.M.H.; Sajadi, S.M. A green multi-objective integrated scheduling of production and distribution with heterogeneous fleet vehicle routing and time windows. *J. Clean. Prod.* **2020**, *259*, 120824. [CrossRef]
16. Xiong, H. Research on cold chain logistics distribution route based on ant colony optimization algorithm. *Discrete Dyn. Nat. Soc.* **2021**, 6623563. [CrossRef]
17. Wang, B.; Peng, X.; Huang, X.; Hu, P.; Su, G.; Yang, Q. The new method for scheduling towing tractors based on distributed strategy. In Proceedings of the 2021 4th International Symposium on Traffic Transportation and Civil Architecture (ISTTCA), Suzhou, China, 12–14 November 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 11–17.
18. Han, X.; Zhao, P.; Kong, D. A bi-objective optimization of airport ferry vehicle scheduling based on heuristic algorithm: A real data case study. *Adv. Prod. Eng. Manag.* **2022**, *17*, 183–192. [CrossRef]
19. Xu, Y.; Fang, M.; Chen, L.; Xu, G.; Du, Y.; Zhang, C. Reinforcement Learning with Multiple Relational Attention for Solving Vehicle Routing Problems. *IEEE Trans. Cybern.* **2021**, *52*, 11107–11120. [CrossRef]
20. Xin, L.; Song, W.; Cao, Z.; Zhang, J. Step-Wise Deep Learning Models for Solving Routing Problems. *IEEE Trans. Ind. Inform.* **2020**, *17*, 4861–4871. [CrossRef]
21. Lei, K.; Guo, P.; Wang, Y.; Wu, X.; Zhao, W. Solve Routing Problems with a Residual Edge-Graph Attention Neural Network. *Neurocomputing* **2022**, *508*, 79–98. [CrossRef]
22. Zou, Y.; Wu, H.; Yin, Y.; Dhamotharan, L.; Chen, D.; Tiwari, A.K. An Improved Transformer Model with Multi-Head Attention and Attention to Attention for Low-Carbon Multi-Depot Vehicle Routing Problem. *Ann. Oper. Res.* **2024**, *339*, 517–536. [CrossRef]
23. Joshi, C.K.; Laurent, T.; Bresson, X. An efficient graph convolutional network technique for the travelling salesman problem. *arXiv* **2020**, arXiv:1906.01227.

24. Fang, Y.; Yang, W.H.; Ihara, Y.; Kamiya, Y. Developing a simple electricity consumption prediction formula for the pre-introduction prediction for electric buses. *World Electr. Vehicle J.* **2025**, *16*, 67. [CrossRef]
25. Kozłowski, E.; Wiśniowski, P.; Gis, M.; Zimakowska-Laskowska, M.; Borucka, A. Vehicle acceleration and speed as factors determining energy consumption in electric vehicles. *Energies* **2024**, *17*, 4051. [CrossRef]
26. Łebkowski, A. Energy consumption optimization for an electric delivery vehicle. *Energies* **2024**, *17*, 5665. [CrossRef]
27. Sundström, O.; Binding, C. Optimization Methods to Plan the Charging of Electric Vehicle Fleets. In Proceedings of the International Conference on Control, Communication and Power Engineering, Chennai, India, 28–29 July 2010; pp. 28–29.
28. Gu, A.; Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv* **2023**, arXiv:2312.00752.
29. Kool, W.; Van Hoof, H.; Welling, M. Attention, learn to solve routing problems! *arXiv* **2018**, arXiv:1803.08475.
30. Williams, R.J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* **1992**, *8*, 229–256. [CrossRef]
31. Kingma, D.P. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
32. Sutton, R.S. *Reinforcement Learning: An Introduction*, 2nd ed.; MIT Press: Cambridge, MA, USA, 2018.
33. Xiamen Airport. Maps. Available online: http://xmairport.cn/lkzn/hjldt.aspx (accessed on 7 February 2025).
34. Xiamen Airport. Flight Information. Available online: https://www.xiamenairport.com.cn/hbxx/hbxx.aspx (accessed on 7 February 2025).
35. Wu, Y.; Zhou, J.; Xia, Y.; Zhang, X.; Cao, Z.; Zhang, J. Neural airport ground handling. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 15652–15666. [CrossRef]
36. Sanz de Vicente, S. Ground Handling Simulation with CAST. Master's Dissertation, Hamburg University of Applied Science, Hamburg, Germany, 2010. Available online: http://www.fzt.haw-hamburg.de/pers/Scholz/arbeiten/TextSanz.pdf (accessed on 23 August 2014).
37. *IBM ILOG CPLEX 12.10 User Manual*; IBM Corp.: Armonk, NY, USA, 2017.
38. Aarons, G.A.; Ehrhart, M.G.; Farahnak, L.R. The implementation leadership scale (ILS): Development of a brief measure of unit level implementation leadership. *Implement. Sci.* **2014**, *9*, 45. [CrossRef]
39. Chen, S.M.; Chien, C.Y. Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. *Expert Syst. Appl.* **2011**, *38*, 14439–14450. [CrossRef]
40. Tian, S.; Chen, H.; Wu, G.; Cheng, J. Asymmetric Arc Routing by Coordinating a Truck and Multiple Drones. *Sensors* **2022**, *22*, 6077. [CrossRef]
41. Lee, Z.J.; Su, S.F.; Chuang, C.C.; Liu, K.H. Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. *Appl. Soft Comput.* **2008**, *8*, 55–78. [CrossRef]
42. Hojabri, H.; Gendreau, M.; Potvin, J.Y.; Rousseau, L.M. Large neighborhood search with constraint programming for a vehicle routing problem with synchronization constraints. *Comput. Oper. Res.* **2018**, *92*, 87–97. [CrossRef]
43. Braysy, O.; Dullaert, W.; Gendreau, M. Evolutionary algorithms for the vehicle routing problem with time windows. *J. Heuristics* **2004**, *10*, 587–611. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.