



OPEN Deep reinforcement learning based low energy consumption scheduling approach design for urban electric logistics vehicle networks

Pengfei Sun^{1,2,3}✉, Jingbo He¹, Jianxiong Wan^{1,2}, Yuxin Guan^{1,2}, Dongjiang Liu^{1,2}, Xiaoming Su^{1,2} & Leixiao Li^{1,2}✉

The rapid increase in carbon emissions from the logistics transportation industry has underscored the urgent need for low-carbon logistics solutions. Electric logistics vehicles (ELVs) are increasingly being considered as replacements for traditional fuel-powered vehicles to reduce emissions in urban logistics. However, ELVs are typically limited by their battery capacity and load constraints. Additionally, effective scheduling of charging and the management of transportation duration are critical factors that must be addressed. This paper addresses low energy consumption scheduling (LECS) problem, which aims to minimize the total energy consumption of heterogeneous ELVs with varying load and battery capacities, considering the availability of multiple charging stations (CSs). Given that the complexity of LECS problem, this study proposes a heterogeneous attention model based on encoder-decoder architecture (HAMEDA) approach, which employs a heterogeneous graph attention network and introduces a novel decoding procedure to enhance solution quality and learning efficiency during the encoding and decoding phases. Trained via deep reinforcement learning (DRL) in an unsupervised manner, HAMEDA is adept at autonomously deriving optimal transportation routes for each ELV from specific cases presented. Comprehensive simulations have verified that HAMEDA can diminish overall energy utilization by no less than 1.64% compared with other traditional heuristic or learning-based algorithms. Additionally, HAMEDA excels in maintaining an advantageous equilibrium between execution speed and the quality of solutions, rendering it exceptionally apt for expansive tasks that necessitate swift decision-making processes.

Keywords Urban electric logistics vehicle networks, Low energy consumption scheduling, Heterogeneous attention model, Deep reinforcement learning

The meteoric expansion of the e-commerce sector has markedly accelerated the evolution of the logistics industry within China. This sector is a significant consumer of conventional fossil fuels and a major contributor to substantial carbon emissions. In light of the progressive degradation of the global climate, the adoption of low-carbon logistics strategies has become an imperative adaptation. According to statistics from Greenpeace Organization¹, carbon emissions associated with China's logistics sector are projected to reach 55.65 million tons in 2022 as shown in Fig. 1, reflecting an average increase of 25.1% over the past six years. The carbon emissions within the logistics industry primarily originate from three major areas: warehousing, packaging materials, and transportation. Notably, transportation alone accounted for 62.7% of the total emissions within the logistics sector in 2022. This highlights the urgent need for low-carbon transportation solutions in logistics, which has spurred innovative research initiatives, including energy-efficient intelligent transportation scheduling², green logistics³ and low-carbon logistics network design⁴.

¹College of Data Science and Application, Inner Mongolia University of Technology, Hohhot 010080, China.

²Inner Mongolia Key Laboratory of Beijing Cyberspace Security, Inner Mongolia University of Technology, Hohhot 010080, China. ³College of Computer Science, Inner Mongolia University, Hohhot 010021, China. ✉email: pfsun@imut.edu.cn; lileixiao@imut.edu.cn

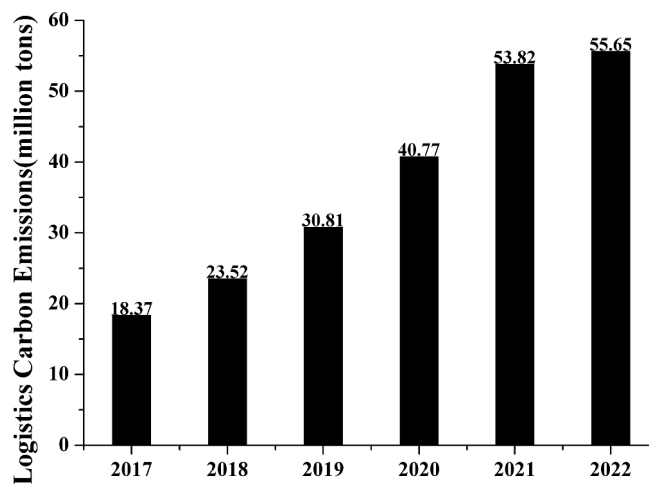


Fig. 1. Logistics business involved carbon emissions in China (2017–2022).

The comprehensive supporting infrastructure and reliable performance of traditional fuel vehicles have established them as the dominant mode of logistics transportation over recent decades. In contrast, electric vehicles (EVs) face several significant challenges, including limited battery capacity, susceptibility to climatic conditions affecting battery performance, and insufficient distribution of charging stations (CSs). It is not uncommon to observe numerous vehicles queuing to charge at these stations. Such issues severely hinder the transportation efficiency within the logistics sector and significantly impede the adoption of EVs in this industry. This is particularly critical in urban logistics, where the timeliness of transportation is a paramount concern. Generally, urban areas tend to have a relatively higher number of CSs. Consequently, a viable strategy to enhance the transportation efficiency of urban logistics involves generating multiple optimal transportation routes for heterogeneous electric logistics vehicles (ELVs) that possess varying load capacities and battery specifications, all while ensuring timely transportation. This approach aims to minimize energy consumption during transit. Each route for an ELV comprises logistics packages (LPs) and CSs, with each vehicle visiting these locations sequentially according to its designated route. Therefore, determining the optimal routes for ELVs for multiple CSs presents a practical and intriguing challenge.

Numerous studies have focused on the formulation of transportation routing optimization models⁵ and the collaborative logistics of pickup and delivery systems^{6,7}. There is also significant research dedicated to the simultaneous optimization of routing and charging for fleets of ELVs⁸. Considering elements such as fixed, transportation, and carbon emission costs, a model for optimizing low-carbon logistics routing for cold chains was developed, integrating constraints concerning vehicle load and delivery windows⁹. Zhang et al.¹⁰ proposed a model for low-carbon, flexible, time-sensitive pickup and delivery (LC-FTSP-TW) aimed at minimizing greenhouse gas emissions within logistics frameworks, accommodating fluctuations in traffic conditions, customer delivery timelines, and vehicle energy requirements. The M-NSGA-II algorithm was employed to reduce energy consumption and carbon emissions in urban logistics transportation¹¹. Considering that logistics transportation must account for factors such as energy consumption and the spatial distribution of CSs, the associated optimization problem exhibits high dimensionality and strong nonlinearity, which often struggle to provide efficient and scalable solutions for traditional optimization methods. Therefore, the deep reinforcement learning (DRL) method featuring an attention mechanism was designed to optimize the longest or total travel duration for vehicles of varying capacities. This method includes a vehicle selection decoder and a node selection decoder, facilitating the automated selection of both a vehicle and a node for that vehicle at each decision point¹². Furthermore, given the constraints of battery capacity, a novel optimal charging strategy for ELVs on long journeys was introduced, providing drivers with precise charging guidance for specific routes¹³. However, none of the aforementioned studies have addressed the heterogeneity of ELVs or the presence of multiple CSs concurrently.

In contrast to existing literature, the proposed LECS system is designed to identify optimal transportation routes for heterogeneous ELVs, each characterized by different load capacities and battery specifications, as shown in Fig. 2. The optimal transportation route is anticipated to be a Hamiltonian tour, representing a permutation of a subset of LPs and CSs. Notably, each LP must be assigned exclusively to the tour of a single ELV and can only be included once. Furthermore, it is assumed that each ELV can achieve a full charge within a reasonable timeframe upon arrival at CS. Unlike traditional logistics scheduling approaches, which primarily focus on vehicle load capacity and routing optimization, the proposed method considers multiple factors, including the heterogeneity of ELVs, the distribution of CSs, and the timeliness of transportation.

The deployment of a LECS holds the promise of substantially curtailing both energy consumption and carbon emissions in urban logistics transportation. This system is particularly appealing to logistics companies that utilize unmanned ELVs. Notably, improvements in charging efficiency could further diminish carbon emissions across the entire urban logistics transportation sector, thereby facilitating the adoption of new energy vehicles from an economic standpoint. However, the challenges associated with LECS system are considerable.

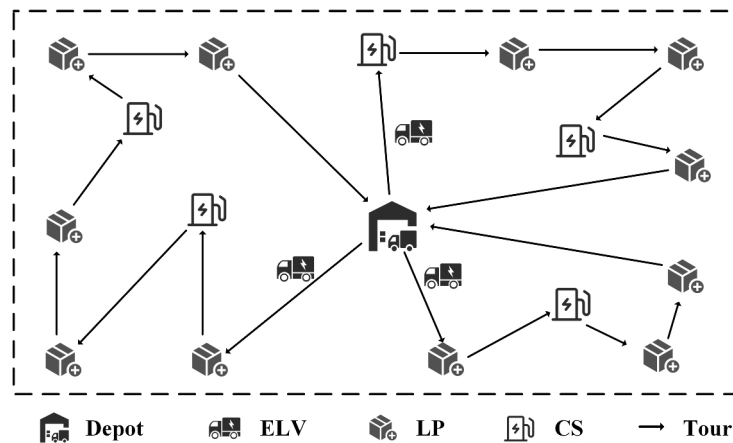


Fig. 2. Illustration of LECS system in urban logistics.

Primarily, it is essential to identify an optimal Hamiltonian transportation tour for each heterogeneous ELV while accommodating multiple CSs and adhering to constraints related to maximum transportation duration. The tour may encompass any subset of all LPs and CSs, indicating that the process of determining the optimal tour may require exponential time for any given ELV. Furthermore, each LP must be assigned precisely once. Due to the high time complexity involved, there currently exists no effective exact algorithm or approximation algorithm capable of addressing LECS problem.

Fortunately, the DRL-based graph attention mechanism dynamically adjusts learnable attention weights, enabling ELVs to focus on critical nodes (such as CSs or key task nodes) at different mission stages while filtering out irrelevant information, which can effectively reduce the interference of noisy nodes and provide efficient and scalable solutions for the associated optimization problem with high dimensionality and strong nonlinearity. Therefore, this paper formulates LECS problem, which aims to minimize the total energy consumption of various heterogeneous ELVs with differing loads and battery capacities, while adhering to the constraints of maximum transportation duration. After analyzing the complexity of LECS problem, it was initially modeled as a Markov decision process (MDP). Subsequently, a heterogeneous attention model based on encoder-decoder architecture (HAMEDA) approach, which is designed to automatically learn a construction policy for solving LECS problem, consisting of an encoding phase and a decoding phase. Finally, HAMEDA is trained using REINFORCE algorithm¹⁴. The primary contributions of this study are outlined as follows:

1. A novel LECS framework is introduced, aiming to optimize total energy usage across a network of diverse ELVs. Through rigorous theoretical examination, it is demonstrated that solving the LECS problem is non-deterministic polynomial-time hard (NP-hard).
2. To capture the complex relationships between LPs and CSs, a heterogeneous graph attention network is employed to encode the features of nodes to the embeddings in the encoding phase of HAMEDA model. During the decoding phase, a masking technique is integrated to effectively manage the graph structure, thereby precluding the selection of infeasible nodes and enhancing the overall solution quality. Furthermore, to enhance training stability, proposed model is trained via DRL in an unsupervised manner with the one-agent-per-decoding-step strategy.
3. Extensive simulations demonstrate that the proposed HAMEDA model reduce the total energy consumption by at least 1.64% compared to traditional heuristic and learning-based algorithms, while only 3.5% higher than the optimal solution. Furthermore, HAMEDA offers an optimal balance between execution time and solution quality, making it particularly suitable for large-scale applications requiring prompt decision-making. The scheme has been open-sourced, enabling researchers to utilize it for evaluating auto-scaling methodologies.

The structure of the remainder of this study is organized as follows: Section II surveys pertinent literature. Section III delineates the system model and articulates the LECS problem. Section IV details the HAMEDA model, proposed as a resolution to the LECS challenge. Section V undertakes a performance assessment. Section VI culminates with the conclusions.

Related work

Low-carbon logistics network design

Recently, the design of networks for low-carbon logistics has garnered significant attention from researchers both domestically and internationally^{15,16}. Liang et al.¹⁷ developed a dynamic generalized method of moments (GMM) model and a Tobit model to investigate the potential temporal and spatial effects of environmental regulation on the green total factor productivity within the low-carbon logistics sector. For regional multimodal logistics networks, Jiang et al.¹⁸ proposed an enhanced adjustable robust optimization method aimed at reducing carbon dioxide emissions. In pursuit of harmonizing the goals of governmental bodies with those of cold chain logistics firms, Zhang et al.¹⁹ devised a bilevel programming-based decision-making model that seeks to minimize the

aggregate costs within the entire cold chain logistics framework. In subsequent research, the intricacies of a low-carbon integrated forward/reverse logistics network were analyzed, leading to the development of a fuzzy stochastic programming model aimed at reducing costs associated with carbon trading balances²⁰. Additionally, to achieve specific carbon dioxide emission targets in a regional logistics network, a bi-level optimization model was developed, integrating the selection of investments in logistics infrastructure and the allocation of subsidies for sustainable transportation modes²¹. To facilitate enterprises in reducing carbon emissions during current logistics operations, Wang et al. formulated a green urban closed-loop logistics distribution network model designed to minimize both greenhouse gas emissions and total operational costs²². Li et al.²³ proposed an optimization model for the EV routing problem, tailored to the sharing economy, with the aim of minimizing comprehensive costs including operational, penalty, queuing, electricity, and environmental expenses. A Catastrophic Adaptive Genetic Algorithm (CA-GA), based on Monte Carlo sampling, was engineered to solve the low-carbon path optimization issue, accounting for transportation costs, time expenses, and carbon emission costs amidst dual uncertainty²⁴. Furthermore, a multistage planning approach for developing low-carbon charging infrastructure for EVs was introduced, featuring an innovative travel route choice model that allows EV drivers to make multiple charging detours before completing their routes²⁵. Zhang et al.²⁶ crafted a sparrow search algorithm enhanced by an adaptive t-distribution for tackling a multi-objective low-carbon multi-modal transportation planning problem with fuzzy demand and fuzzy time (MOLCMTTP-FDFT), aiming to minimize both costs and time while integrating essential policies on carbon emissions, carbon taxes, carbon trading, and carbon offsets.

The aforementioned methods address the challenges associated with low-carbon path optimization and decision-making in dynamic network environments. In contrast to the studies cited above, this paper introduces an innovative scheduling mechanism aimed at minimizing energy consumption and carbon emissions produced by heterogeneous ELVs during urban logistics transportation.

Cost optimization method design

Challenges within the transportation sector are broadly categorized under generalizations of the vehicle routing problem (VRP) or the traveling salesman problem (TSP), both recognized as NP-hard issues. Consequently, conventional strategies to manage such NP-hard challenges generally fall into three categories: exact algorithms, heuristic algorithms, and methods reliant on DRL. Exact algorithms are designed to yield optimal solutions through enumeration or other techniques. However, they generally exhibit exponential running times²⁷. In contrast, approximate algorithms can deliver feasible solutions within polynomial time and are capable of identifying approximate solutions that are relatively close to the optimal for large-scale cost optimization challenges^{28,29}. Compared to the aforementioned two categories, DRL-based methods tend to produce solutions more rapidly and possess the ability to autonomously learn representations of states and actions. This characteristic diminishes the dependence on domain-specific knowledge and may uncover complex patterns that traditional methods fail to capture. To derive an approximate solution for covering salesman problem (CSP), Li et al.³⁰ introduced a novel deep learning approach utilizing multi-head attention (MHA), which was trained using unsupervised DRL techniques. To reduce the overall flight distance of unmanned aerial vehicles (UAVs), this study introduces a DRL-based approach, employing a multi-head heterogeneous attention (MHHA) mechanism. This mechanism supports the strategic sequential formulation of routes with an emphasis on energy efficiency³¹. Several frameworks leveraging DRL have been developed to address a challenging yet nontrivial variant of TSP^{32,33}. In addressing large-scale problem scenarios to enhance solution quality, Zhao et al. suggested a novel DRL model, which incorporates a local search method aimed at further elevating the quality of solutions. This model is structured around an actor, an adaptive critic, and a routing simulator³⁴. Additionally, an adaptive car-following trajectory control algorithm (e.g., deep adaptive control) was developed to address the challenges associated with adaptive vehicle trajectory control across varying risk levels³⁵. Furthermore, to decrease the total charging time for EVs, the research presented a DRL algorithm focused on reducing not only the overall charging duration for EVs but also achieving significant reductions in the origin-destination distance³⁶.

However, the presence of heterogeneous vehicles and the mixing of CS and LP nodes can negatively affect the convergence and efficiency of optimization models, factors that have not been adequately addressed by the previously mentioned methods. In contrast to the works discussed above, this paper proposes a heterogeneous attention model-based approach HAMEDA for LECS problem utilizing DRL. This approach demonstrates superior performance compared to traditional heuristic methods and other learning-based techniques.

System model and problem formulation

System model

This study examined a low-energy consumption scheduling system composed of a logistics cloud platform, an online community consisting of n users and z CSs, as well as m ELVs with varying load and battery capacities. Each user is associated with an LP that requires collection by an ELV. For clarity, $N = \{1, 2, \dots, n\}$ is used to uniformly represent both users and their corresponding LPs. $Z = \{1, 2, \dots, z\}$ is employed to denote the collection of CSs, while $M = \{1, 2, \dots, m\}$ is used to represent ELVs. All computations for this system are conducted by the logistics cloud platform. Given the presence of multiple CSs, each may be visited by an ELV multiple times or not at all. It is assumed that these CSs have been established within the region of interest, and the location information $(x_0^e, x_1^e, \dots, x_z^e)$ of CSs has been provided to the platform. Each user $i \in N$ is also required to submit pertinent information $UI_i = (w_i^{LP}, x_i^u)$ to the platform, which includes w_i^{LP} and x_i^u , representing the weight of LP i and its current location, respectively. Generally, weight serves as a standard criterion for logistics services³⁷. Additionally, each ELV must submit information $VI_m = (x_m^v, R_m, R_m^r, r_m, w_m^{\max}, w_m^{ELV})$ to the platform, where x_m^v is the current location of ELV m and R_m, R_m^r, r_m are the maximum battery capacity, current battery capacity, and unit battery consumption of ELV, respectively. It is assumed that each CS possesses

unlimited energy capacity, ensuring that all arriving ELVs can be fully recharged. w_m^{\max} , w_m^{ELV} are the maximum and current load weight of ELV m 's, respectively.

LECS system represents a complete graph $G = (X, E)$, whereis a combination of LP set $X^{LP} = (x_1, \dots, x_n)$, the depot x_0 , and CSs set $X^{CS} = (x_{n+1}, \dots, x_{n+z})$. The node setcan be constructed as $X = \{x_0\} \cup X^{CS} \cup X^{LP}$. $E = (\{x_i, x_j\} : i < j)$ is the edge set. Each edge $\{x_i, x_j\}$ is associated with a non-negative travel time $t_{i,j}$ and the shortest route length $d(x_i, x_j)$.

A feasible ELV m 's tour π^m commences at the depot x_0 , sequentially traverses a series of LPs and CSs, and ultimately returns to x_0 , where each CS may be accessed multiple times. To facilitate the representation of ELV access sequence in the most efficient manner, this study introduce $\delta_z - 1$ dummy nodes for each CS, which share their respective locations. The set of dummy nodes is X_d^{CS} . The number δ_z of dummy nodes associated with each CS, is determined by the frequency of visits to the corresponding CS z . It is advisable to minimize δ_z to reduce the overall size of the network³⁸. For convenience, $I = X^{CS} \cup X^{LP} \cup X_d^{CS}$ and $Z' = X^{CS} \cup X_d^{CS}$ is defined. Consequently, the total number of nodes in the resulting augmented graph $G' = (I, E)$ can be calculated as follows:

$$count = n + 1 + \sum_{k=0}^z \delta_z$$

(1)

The following section presents the formulation of the problem and an analysis of its complexity. Table 1 provides a list of commonly used notations.

Problem formulation

In an LECS system, the study aims to identify a maximum of transportation Hamiltonian tours, one for each ELV, that commence and conclude at the depot while visiting a selected subset of nodes, including CSs as necessary, to minimize the total energy consumption associated with transportation. This study defines the arc set $A = \{(x_i, x_j) : i < j \text{ and } i, j \in I\}$ within graph G' . Therefore, the edge set E is substituted with an arc set A , where each arc $(x_i, x_j) : i < j \text{ and } i, j \in I$ corresponds to a respective edge $\{x_i, x_j\}$. Additionally, the decision variable $x_{i,j}$ is expressed as follows:

$$x_{i,j} = \begin{cases} 1 & \text{if } x_i, x_j \in \pi^m, i < j \\ 0 & \text{otherwise} \end{cases}$$

(2)

The duration of the transportation tour for ELV's can be expressed as follows:

Symbol	Description
N, n	Set of LPs, number of LPs
Z, z	Set of CSs, number of CSs
M, m	Set of ELVs, number of ELVs
$U I_i$	Information submitted by the user i
$V I_m$	Information submitted by ELV m
x_i^u	User i 's location
x_m^v	ELV m 's location
w_i^{LP}	LP i 's weight
w_m^{\max}	ELV m 's maximum load capacity
R_m	ELV m 's maximum or current battery capacity
R_m^r	ELV m 's remaining battery capacity
r_m	ELV m 's unit energy consumption
w_m^{ELV}	ELV m 's remaining load capacity
$G (X, E)$	The nodes set and edge set in graph G
$count$	Total number of nodes in G'
x_0, x_{n+z+1}	Depot
$d(x_i, x_j)$	The shortest distance between x_i and x_j
π^m	Optimal pickup Hamiltonian tour of ELV m
$dist(\pi_m)$	Route length of ELV m
$Q_m (\pi_m)$	The total battery consumption of ELV m
T_{\max}	Maximum duration
τ_i	The time of arrival of a ELV at node i
η_j	Service or charging time

Table 1. Frequently used notations.

$$dist(\pi^m) = \sum_{x_i, x_j \in \pi^m, i < j} d(x_i, x_j) \cdot x_{i,j} \quad (3)$$

where $d(x_i, x_j)$ is the shortest route length between node x_i and node x_j .

The remaining load capacity of ELV m is expressed as follows:

$$w_m^{ELV} = w_m^{\max} - \sum_{i \in \pi^m} w_i^{LP} \quad (4)$$

The remaining transportation duration of ELV m is given as follows:

$$T_m^{ELV} = T_{\max} - \sum_{j \in \pi^m} \eta_j - \sum_{i, j \in \pi^m, i < j} t_{i,j}(m) x_{i,j} \quad (5)$$

and

$$t_{i,j}(m) = d(x_i, x_j)/v_m \quad (6)$$

where T_{\max} is the maximum transportation duration for each EV, which is treated as a constant; $t_{i,j}$ is the travel time from node i to node j ; v_m is the average speed of ELV m . η_j is also assumed to be constant, signifying the service time when node i is classified as a LP; otherwise, it indicates the recharging time.

The total energy consumption of ELV m can be expressed as follows:

$$Q_m(\pi^m) = r_m \cdot dist(\pi^m) \quad (7)$$

The remaining battery capacity of ELV m is given as follows:

$$R_m^r = R_m - Q_m(\pi^m) \quad (8)$$

The objective of this study is to minimize the total energy consumption of all ELVs. This issue is defined as LECS problem, which can be articulated as follows:

$$(LECS) \quad \min \sum_{m \in M} Q_m(\pi^m) \quad (9)$$

$$s.t. \quad \sum_{j: (x_i, x_j) \in A} x_{i,j} = \sum_{j: (x_j, x_i) \in A} x_{j,i} = 1, \forall i \in X^{LP} \quad (9-1)$$

$$\sum_{j: (x_i, x_j) \in A, i \neq j} x_{i,j} \leq 1, \forall i \in X \quad (9-2)$$

$$\sum_{j: (x_i, x_j) \in A, i \neq j} x_{i,j} = \sum_{j: (x_j, x_i) \in A, i \neq j} x_{j,i}, \forall i \in I \quad (9-3)$$

$$\sum_{i \in I, i \neq j} x_{i,0} = \sum_{i \in I, i \neq j} x_{0,i} \leq m, \forall j \in M \quad (9-4)$$

$$\sum_{i \in \pi^m} w_i^{LP} \leq w_m^{\max}, \forall m \in M \quad (9-5)$$

$$R_m - r_m \cdot d(x_i, x_j) \cdot x_{i,j} \geq 0, \forall m \in M, i \in N, j \in Z' \quad (9-6)$$

$$R_m - r_m \cdot d(x_i, x_j) \cdot x_{i,j} \geq 0, \forall m \in M, i \in N, j \in Z' \quad (9-7)$$

$$\pi^m \cap \pi^{m'} = \{x_0\}, \forall m, m' \in M, m \neq m' \quad (9-8)$$

$$s_i - s_j + 1 \leq (1 - x_{i,j}) |\pi^m|, \forall i, j \in \pi^m, i \neq j \quad (9-9)$$

$$1 \leq s_i \leq |\pi^m|, \forall i \in \pi^m \quad (9-10)$$

$$\tau_j \geq \tau_i + (t_{i,j} - \eta_j) x_{i,j} - T_{\max}(1 - x_{i,j}), \forall i \in I, j \in I \setminus \{0\} \quad (9-11)$$

$$0 \leq \tau_0 \leq T_{\max} \quad (9-12)$$

$$t_{0,j} \leq \tau_j \leq T_{\max} - (t_{j,0} + \eta_j), \forall j \in I \setminus \{0\} \quad (9-13)$$

$$x_{i,j} \in \{0, 1\}, \forall i \in I \quad (9-14)$$

where τ_i is a time variable that indicates the arrival time of an ELV at node i , which is initialized to zero upon departure from the depot. Constraint (9-1) stipulates that each LP must be accessed only once. Constraint (9-2) ensures that each CS is visited at most once, including the dummy nodes. Constraint (9-3) mandates that each LP possesses a single in-degree and a single out-degree. Constraint (9-4) limits the total number of tours to a maximum of m . Furthermore, each ELV is required to commence and conclude its journey at the depot. Constraint (9-5) guarantees that the cumulative weight of all LPs within ELV m 's tour π_m does not exceed its maximum load capacity. Constraint (9-6) ensures that each ELV can access the subsequent LP based on its current battery capacity. Similarly, constraint (9-7) confirms that each ELV can reach the nearest CS, also contingent upon its current battery capacity. Constraint (9-8) prohibits the overlap of routes between any two ELVs, with the exception of the depot. Constraints (9-9) and (9-10) prevent the formation of subtours within the route of any ELV. The arrival time at each node for each ELV is monitored through constraint (9-11). Constraint (9-12) establishes an upper limit on arrival times upon return to the depot, specifically the maximum transportation duration T_{\max} . Constraints (9-13) provide lower and upper bounds on arrival times at LPs and CSs, ensuring that each tour is completed by time T_{\max} . The final constraint defines the variables as binary.

Hardness analyzing

This study endeavors to identify an exact algorithm for LECS problem. However, as demonstrated by Theorem 1, LECS problem is classified as NP-hard. Subsequently, the complexity associated with LECS problem is investigated.

Theorem 1 LECS problem is NP-hard.

In the realm of logistics optimization, the Green vehicle routing problem (G-VRP)³⁹ is characterized on a comprehensive graph wherein the vertices symbolize the locations of LPs, ELVs, and a central depot. The primary aim of G-VRP is to ascertain a collection of routes for ELVs that optimize efficiency by minimizing the cumulative distance traveled. Each tour commences at the depot, visits a designated set of LPs within a predetermined time constraint, and subsequently returns to the depot, all while adhering to the driving range limitations imposed by the battery capacity of ELVs. Additionally, each tour may incorporate stops at one or more CSs to facilitate recharging of ELVs during the journey. G-VRP can be expressed as follows:

$$\begin{aligned} (G - VRP) \quad & \min \sum_{m \in M} dist(\pi^m) \\ s.t. \quad & (9 - 1) - (9 - 4) \text{ and } (9 - 6) - (9 - 14) \end{aligned} \quad (10)$$

LECS problem presents differences from G-VRP in two main aspects. Firstly, LECS incorporates an additional constraint (9-1) that ensures the heterogeneity of ELVs with varying load weights. Furthermore, the objective function (9) of LECS considers the heterogeneity of ELVs in terms of their different unit energy consumption. Therefore, the G-VRP problem can be seen as the special case of LECS problem, where the ELVs' load weights are unfinite. Obviously, this problem is a generalization of the well-known NP-hard G-VRP problem³⁹. This indicates that LECS problem is at least as complex as G-VRP. Given that G-VRP is classified as NP-hard³⁹, it follows that LECS problem is also NP-hard.

Reformulation as MDP

Given that LECS problem is classified as NP-hard, obtaining an optimal solution within polynomial time is infeasible. Route formulation within the LECS is envisioned as a series of sequential decision-making tasks. This perspective allows for a natural formulation of the problem as an MDP model, which can be effectively addressed using Encoder-Decoder architecture⁴⁰.

1) State: The global state S_t encompasses comprehensive state information, whereas each ELV m possesses only partial state information $s_t^m, s_t^m \in S_t$. The state $s_t^m = (\pi_t^m, R_m^r(t), w_m^r(t), T_m^r(t))$ comprises the partial solution π_t^m , the remaining battery capacity $R_m^r(t)$, the remaining load capacity $w_m^r(t)$, and the remaining duration limit $T_m^r(t)$ at time step t for ELV m . Additionally, π_t^m contains all nodes that have been visited up to time step t .

2) Action: The action space at time step t is $a_t^m = (N_t^r, Z_t)$, $m \in M$, where N_t^r is the set of available LP nodes that have not yet been visited but can be accessed by ELV m at time step t , Z_t is the set of available CS nodes that ELV m can reach at time step t . Therefore, the action space for all ELVs can be defined as follows:

$$A_t = \bigcup_{m \in M} a_t^m \quad (11)$$

The agent must select an action A_t from the set at time step t unless it reaches a terminal state.

3) Reward: To minimize the overall energy consumption of all ELVs, the reward is defined as the negative of the objective value. Consequently, the reward function R is computed as follows:

$$R = - \sum_{m \in M} Q_m(\pi^m) \quad (12)$$

4) Transition: The subsequent state $s_{t+1}^m = (\pi_{t+1}^m, R_m^r(t+1), w_m^r(t+1), T_m^r(t+1))$ is determined by the node selected at time step t . The partial solution is concatenated with the newly selected node j (i.e., resulting

in $\pi_{t+1}^m = (\pi_t^m; \{j\})$. The modifications to $w_m^r(t+1)$ and $T_m^r(t+1)$ are executed in accordance with Eq. (4) and (5). Specifically, if node j is classified as a LP node, $R_m^r(t+1) = R_m^r(t) - r_m(i, j)$ is defined, where $r_m(i, j)$ is the energy consumption of ELV m for traversing from the previous node i to the current node j . Conversely, if node j is categorized as a CS node, then $R_m^r(t+1) = R_m^r$ is applicable.

Methods

This section introduces HAMEDA model as a solution to LECS problem, which facilitates the dynamic extraction of additional features from nodes. The HAMEDA model, upon completion of training, facilitates the calculation of the optimal value of the objective function. This calculation leverages the parameters derived from training and specific input instances. As illustrated in Figure 3, the HAMEDA model employs an Encoder-Decoder architecture to parameterize and educate the agent's policy $pro_{\theta}(\pi|X)$. Each decision-making round is divided into T time steps. At each time step t , the encoder projects and embeds the graph nodes, capturing their spatial and contextual relationships, while the one-agent-per-decoding-step procedure is employed to improve the creation of optimal transportation routes for the decoder. Specifically, this policy strategically selects nodes sequentially for each ELV, culminating in a node permutation that integrates the depot, all LPs, and a subset of CSs, collectively denoted as $\pi = \pi^0, \dots, \pi^m, \dots, \pi^{|M|}, m \in M$. Furthermore, is a permutation of the selected nodes for ELV m , which can be defined as follows:

$$\pi^m = \pi_0^m (= x_0), \pi_1^m, \dots, \pi_{|\pi^m|}^m (= x_0) \quad (13)$$

where $|\pi^m|$ is the total number of nodes selected by ELV m .

Consequently, for the input instance X and the set of learnable parameters θ , the probability distribution $pro_{\theta}(\pi|X)$ of solution π is regarded as the policy for determining π given θ . The policy function can be expressed as follows:

$$pro_{\theta}(\pi|X) = \prod_{m \in M} \prod_{t=1}^{|\pi^m|} pro_{\theta}(\pi_t^m | X, \pi_{1:t-1}^m) \quad (14)$$

where π_t^m is the node selected by EV at decision step t ; $\pi_{1:t-1}^m$ is the node selected by m at $t-1$ time steps. The objective is to identify the optimal parameter set θ^* that generates the optimal tour π^* for EVs.

Model design

The HAMEDA framework is bifurcated into two pivotal phases: encoding and decoding. The encoding phase sees the encoder extracting structural features and embedding each graph node of the input instance X . These embeddings are crucial for generating the key of HAMEDA. During the decoding phase, the framework

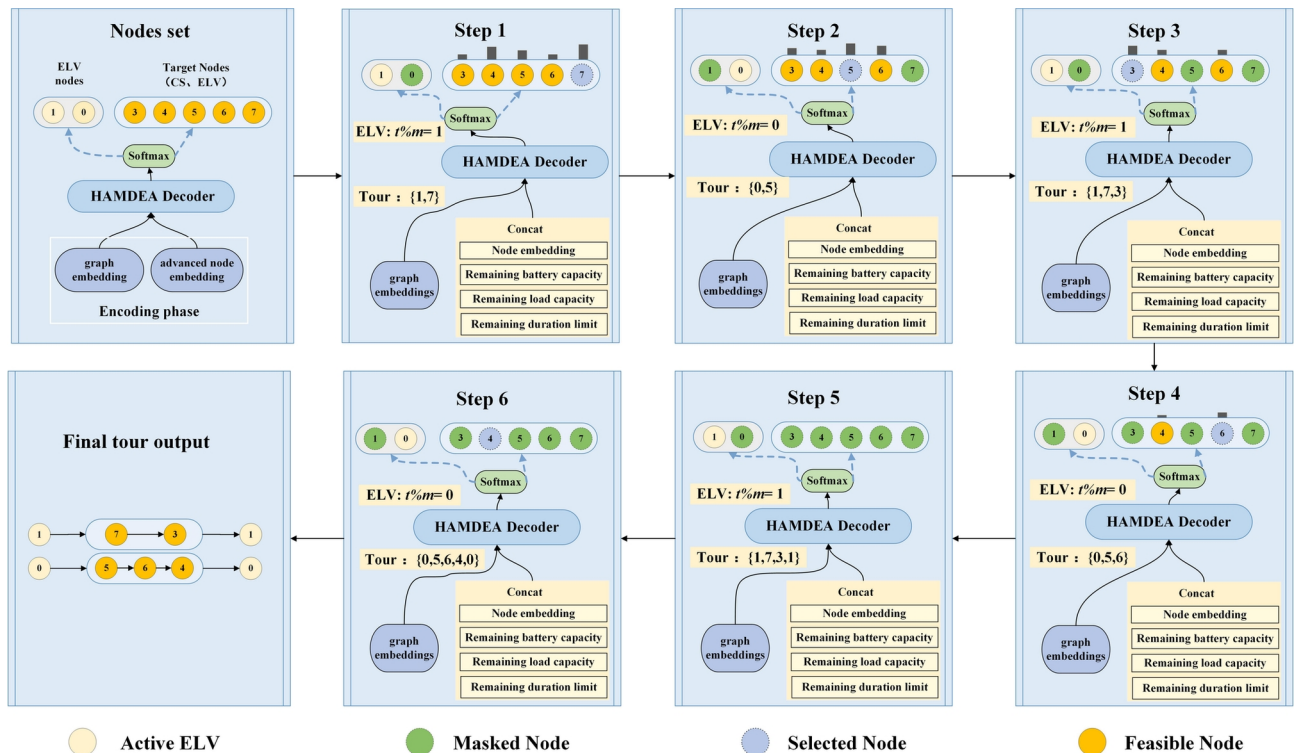


Fig. 3. Tour construction of HAMEDA model applied to two ELVs and five target nodes.

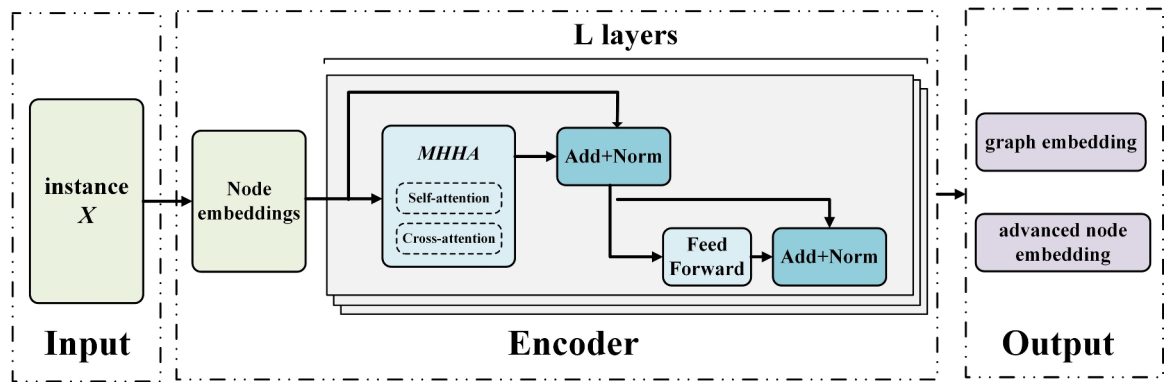


Fig. 4. Illustration of encoding phase in HAMEDA model.

Input: Training dataset X , batch size B , number of epochs N_{epoch} , number of steps N_{steps} per epoch.

Output: Trained parameters θ .

```

1: Initialize model parameters  $\theta, \theta^*$  or policy and baseline network and set  $\theta^* \leftarrow \theta$ ;
2: for each  $epoch \leftarrow 1 : N_{epoch}$  do
3:   for each  $step \leftarrow 1 : N_{steps}$  do
4:      $\lambda \leftarrow RandomSelected(X)$  for  $k \in \{1, 2, \dots, B\}$ ;
5:      $\pi_i \leftarrow pro_{\theta}(\pi|\lambda)$  by sampling;
6:      $\pi_i^* \leftarrow pro_{\theta^*}(\pi|\lambda)$  by greedily;
7:     if  $R(\pi_i) - R(\pi_i^*) \geq 0$  then
8:        $d_{\theta} = \frac{1}{B} \sum_{\lambda=1}^B (R(\pi_i) - R(\pi_i^*)) \cdot \nabla_{\theta} pro_{\theta}(\pi|\lambda)$ ;
9:       update  $\theta$  according to Eq.35;
10:    else
11:      pass;
12:    end if
13:  end for
14:  if  $OneSidePairedTTest(pro_{\theta}, pro_{\theta^*}) < \alpha$  then
15:     $\theta^* \leftarrow \theta$ ;
16:  end if
17: end for
18: return  $\theta$ ;

```

Algorithm 1. Training HAMEDA using REINFORCE.

processes these embeddings along with the step context and a mask denoting infeasible nodes, thereby aiding the node selection process for each ELV. Given the requirement to manage transportation tours for numerous ELVs concurrently, this study introduces an innovative decoding strategy tailored to the multi-agent dynamics of the problem. This strategy effectively mitigates the challenges posed by the combinatorial expansion inherent in the multi-agent action space.

Encoding phase

To construct the *key* for each node, a heterogeneous graph attention (HGA) network-based encoder is employed to encode the features of the nodes into embeddings that encapsulate the structural patterns of the input instance X . As illustrated in Figure 4, the encoder comprises L identical attention layers. Each attention layer consists of two sub-layers: a multi-head heterogeneous attention (MHHA) sublayer and a simple, position-wise feed-forward (FF) sublayer. MHHA sublayer incorporates both self-attention and cross-attention mechanisms, enabling the identification of relationships among all nodes, including CS nodes and ELV nodes. FF sublayer functions as a fully connected network. A residual connection operation is implemented between every two sublayers, followed by a normalization operation. The d_x -dimensional input node x_i is initially mapped to the d_h -dimensional initial node embeddings $h_i^{(0)}$. This mapping can be achieved through a linear transformation that incorporates learnable parameters as follows:

$$h_i^{(0)} = \begin{cases} Wx_i + b & \text{if } i \neq 0 \\ W_0x_i + b_0 & \text{if } i = 0 \end{cases} \quad (15)$$

where $W, W_0 \in \mathbb{R}^{d_x \times d_h}$ and $b, b_0 \in \mathbb{R}^{d_h}$ are the learnable parameters; and x_0 is the starting node when $i = 0$ is considered. Consequently, this study employs the self-reliant parameters W_0, b_0 for x_0 . Given that there are $count - 1$ nodes in graph G' , excluding the depot, the configuration is denoted as $(count - 1) d_h$ for the initial node embedding.

To extract features, the self-attention mechanism is employed to compute the *key*, *query*, and *value* vectors for node i , which can be expressed as follows:

$$\begin{aligned} K_i &= W^k h_i^{(L-1)} \\ q_i &= W^q h_i^{(L-1)} \\ v_i &= W^v h_i^{(L-1)} \end{aligned} \quad (16)$$

where $W^k, W^q \in \mathbb{R}^{d_h \times d_q}$ and $W^v \in \mathbb{R}^{d_h \times d_v}$ are the learnable weight parameters; while $d_q = d_v = d_h / \varpi$ and ϖ are the number of attention heads. This configuration enables the model to extract a greater number of structural features.

The scaled dot product of the *query* vector of node i and the *key* vector of node j is determined to assess the compatibility between the two nodes. This computation can be expressed as follows:

$$u_{i,j} = \frac{q_i^T K_j}{\sqrt{d_q}} \quad (17)$$

The node weight is subsequently determined by applying a softmax function, as outlined below:

$$a_{i,j} = \text{softmax}(u_{i,j}) = \frac{e^{u_{i,j}}}{\sum_{j'=1}^n e^{u_{i,j'}}} \quad (18)$$

The described methodology facilitates the elucidation of relationships between any two nodes, with particular emphasis on the interaction between charging nodes and their target counterparts. Consequently, the cross-attention mechanism is employed to extract additional structural features. In this model, the *key* K_j^c , *query* q_i^e , and *value* v_j^c vectors can be articulated as follows:

$$\begin{aligned} K_j^c &= W^{kc} h_j^c \\ q_i^e &= W^{qe} h_i^e \\ v_j^c &= W^{vc} h_j^c \end{aligned} \quad (19)$$

where h_i^e and h_j^c are the embeddings of LP node i and CS node j , respectively. All parameters W^{kc}, W^{qe}, W^{vc} are subject to training. Therefore, the weights assigned to the connections between CS nodes and LP nodes are computed as follows:

$$u_{i,j}^{ec} = \frac{(q_i^e)^T K_j^c}{\sqrt{d_q}} \quad (20)$$

$$a_{i,j}^{ec} = \text{softmax}(u_{i,j}^{ec}) = \frac{e^{u_{i,j}^{ec}}}{\sum_{j'=1}^n e^{u_{i,j'}^{ec}}} \quad (21)$$

This study leverages the cumulative output of all attention heads to enhance the single head vector $h_i^{(\varpi)}$, which can be formally expressed as follows:

$$h_i^{(\varpi)} = \sum_j a_{i,j} v_j + \sum_j a_{i,j}^{ec} v_j^c \quad (22)$$

Notably, while CS nodes might be visited repeatedly, LP nodes are restricted to a single visit. Consequently, the attention mechanisms from LP nodes to CS nodes primarily enhance the embeddings of the LP nodes alone. In contrast, the attention heads associated with CS node embeddings yield a value of zero. Following the concatenation of messages from various heads, the resulting multi-head vector is processed through a skip-connection layer and a batch normalization (BN) layer as follows⁴¹:

$$h_i^{t'} = \text{BN}(h_i^{(L-1)} + W^{out} \cdot \text{concat}\{h_i^{(1)}, \dots, h_i^{(\varpi)}\}) \quad (23)$$

where $\text{concat}\{\cdot\}$ is the concatenation operator; $BN(\cdot)$ is BN function; and W^{out} is a trainable parameter. Subsequently, the output vector from FF sublayer is transmitted through a skip connection and a BN layer, as described below:

$$h_i^l = BN(h_i^{l'} + FF(h_i^{l'})) \quad (24)$$

Through the progressive traversal across L attention layers, refined node embeddings are obtained. Subsequently, a comprehensive graph embedding is calculated, encapsulating overarching graph information.

$$\overline{h^N} = \sum_{i \in X} h_i^L / n + 1 + z \quad (25)$$

Both the graph embeddings $\overline{h^N}$ and the advanced node embeddings h_i^L are utilized as inputs to the decoder.

Decoding phase

During the encoding phase, leveraging both the refined node embeddings and the comprehensive graph embedding, along with the intermediate solution $\pi_{1:t-1}^m$ at each construction step $t \in \{1, \dots, T\}$, this study ascertains the subsequent node π_t^m in the decoding phase. The decoder, at each step, creates a probability distribution over the nodes, informed by the embeddings generated during the encoding phase. However, if multiple ELVs simultaneously make decisions, the dimensionality of the joint action space increases exponentially, leading to a substantial rise in training complexity and computational overhead. To address this challenge, we adopt the one-agent-per-decoding-step strategy, ensuring that only one agent makes a decision per time step. This approach effectively transforms the problem into a sequential optimization process, progressively constructing an optimal solution, thereby improving the stability for creation of optimal transportation routes. A pivotal initial step involves identifying which ELV will make decisions at each time step t as is shown in Figure 3. Thereafter, the current active index ℓ of the ELV is established as follows:

$$\ell \leftarrow t \% m \quad (26)$$

A context vector $h_\ell^{ct}(t)$ is determined at each decoding step t , incorporating the global graph embedding $\overline{h^N}$, the embedding of the most recently visited node, and the remaining battery capacity $R_m^r(t)$, the remaining load capacity $w_m^r(t)$, and remaining time duration $T_m^r(t)$. For the initial step of decoding, the embedding of the last visited node is replaced with trainable parameters. The computation of the context vector $h_\ell^{ct}(t)$ is detailed below:

$$h_\ell^{ct} = W^{graph} \cdot \overline{h^N} + W^{step} \cdot \text{concat}\{Input_t, R_m^r(t), w_m^r(t), T_m^r(t)\} \quad (27)$$

where

$$Input_t = \begin{cases} W^d & \text{if } t = 1 \\ h_{\pi_{\ell}^{\ell}}^L & \text{if } t > 1 \end{cases} \quad (28)$$

where W^{graph} , W^d and W^{step} are the matrices of trainable parameters.

In the process of computing node selection π_t^ℓ , it is essential to take into account not only the previously selected nodes $\pi_{1 \sim t-1}^\ell$ but also the compatibility between these selected nodes and the remaining nodes. This study ensures a logical and efficient progression in the selection process, optimizing the overall route based on proximity and compatibility criteria π_t^ℓ . Therefore, in light of these considerations, an additional attention mechanism⁴² is incorporated to process h_ℓ^{ct} , which facilitates the acquisition of more comprehensive information.

The query q_ℓ is constructed as follows:

$$q_\ell = \text{soft max}\left(\frac{h_\ell^{ct} K_1^T}{\sqrt{d_k}}\right) V_1 \quad (29)$$

where K_1 and V_1 are linear projections of the node embeddings h_ℓ^{ct} . d_k is the scaling factor and $d_k = \frac{d_h}{M}$, with M denoting the number of heads in Scaled Dot-Product Attention mechanism⁴³.

The key vector for node i in the single-head attention mechanism is computed as follows:

$$k_i = W^K h_i^L \quad (30)$$

Through the aforementioned operation, the query-key (q_ℓ, k_i) is derived. The compatibility between the query and the key is subsequently calculated according to the following methodology.

$$u_i^\ell = \frac{q_\ell^T k_i}{\sqrt{d_k}} \quad (31)$$

During network training, the presence of infeasible nodes in the action space increases the complexity of training and hinders convergence. The masking process improves learning efficiency by dynamically filtering out infeasible nodes to reduce the exploration space, thereby accelerating the convergence of the optimal strategy and enhancing the overall solution quality. Therefore, u_i^ℓ is constrained between $-R$ and $+R$ utilizing the $\tanh(\cdot)$ function, while the infeasible nodes are masked. In this masking process, the infeasible nodes are assigned a value of $-\infty$ to facilitate the updating of u_i^ℓ .

$$u_i^\ell \leftarrow \tanh(u_i^\ell) \cdot R \quad (32)$$

The infeasible nodes consist of the following categories:

- 1) LP nodes that have been already visited;
- 2) LP nodes that whose package weight exceed remaining load weight of current ELV;
- 3) The LP or CS nodes that the ELV cannot reach at the current step due to the insufficient battery capacity or insufficient remaining duration;
- 4) LP nodes that would cause the ELV to fail to reach any CS nodes at the next step if the ELV visits them at the current step;
- 5) The depot when there are still remaining target nodes that need to be visited.

To facilitate the normalization of final probabilities within the range $[0, 1]$, this study utilizes the softmax function. Thus, the likelihood of choosing node i at the decoding step t is formulated as follows:

$$pro_i^t = \text{softmax}(u_i^\ell) = \frac{e^{u_i^\ell}}{\sum_{j=1}^n u_j^t}, i = 1, \dots, n \quad (33)$$

Assuming the application of a greedy strategy to ascertain the pickup tour π^ℓ , this study selects the node with the highest probability p_i^t as π_t^ℓ during the decoding step t . Specifically, the decoder generates a probability distribution across all nodes as described by Eq. (33). Subsequently, a node is selected to visit based on a specified strategy and append it to the end of $\pi_{1:t-1}^\ell$ at each construction step. This decoding procedure repeats iteratively until all LP nodes are visited, and ELVs return to the depot.

Model training

HAMEDA model is trained to obtain parameters θ through policy gradient methods utilizing REINFORCE algorithm¹⁴. This approach integrates both the policy network and the baseline network. The policy network $pro_\theta(\pi|X)$ generates a probability distribution over the action space based on the current state. The baseline network functions to provide a standard reward, aiming to decrease variance with a greedy rollout strategy that selects the action of highest probability. Notably, the baseline network shares an identical architecture with the policy network. Subsequently, this study employs gradient descent to update θ , and the gradients of θ can be expressed as follows:

$$d_\theta = \frac{1}{B} \sum_{\lambda=1}^B (R(\lambda) - b(\lambda)) \cdot \nabla_\theta pro_\theta(\pi|\lambda) \quad (34)$$

$$\theta \leftarrow \text{Adam}(\theta, d_\theta) \quad (35)$$

where B is the batch size and $b(\cdot)$ is a baseline function within the baseline network. The implementation of the baseline function facilitates a significant reduction in computational costs and enhances the rate of convergence. θ can be updated utilizing the Adam optimizer⁴⁴.

The training algorithm employed in the proposed method is delineated in **Algorithm 1**. The baseline function $b(\cdot)$ is instantiated as $R(\pi_i^*)$, which is developed through a greedy rollout approach, while $R(\pi_i)$ is constructed via a sample rollout method. Initially, it is essential to initialize the model parameters θ , θ^* with random values for both the policy and the baseline policy, concurrently setting $\theta^* \leftarrow \theta$ (line 1). Subsequently, HAMEDA model undergoes training based on predetermined epochs and steps (lines 2-16). During the training process, instances are randomly selected from the sample set X (line 4), and the policy is executed through sampling and greedy methods to compute the transportation tours π_i and π_i^* , respectively (lines 5-6). When the sampled solution π_i demonstrates superior performance compared to the greedy solution π_i^* , Monte Carlo sampling is employed to iteratively refine the parameters, thereby enhancing the policy model parameters as delineated in lines 7-10, where B represents the batch size. Additionally, a paired t-test is performed at each epoch to determine whether updates are necessary for the baseline parameters, as specified in lines 13-15. Should the policy outperform the baseline, the parameters of the latter are replaced with those of the former. After extensive training iterations within each epoch, the refined policy network is capable of generating effective solutions.

Once HAMEDA model has been trained using **Algorithm 1**, the optimal Hamiltonian transportation tour π for ELVs can be calculated based on the trained parameters θ and the given input instance X .

Parameters	Value	Parameters	Value
B	512	N_{epoch}	500
N_{steps}	500	d_x	2
optimizer	Adam	learning rate	0.0001
ϖ	8	L	3
d_k	64	d_h	128
decaying rate	0.995	α	0.05

Table 2. Default parameter settings for HAMEDA model.

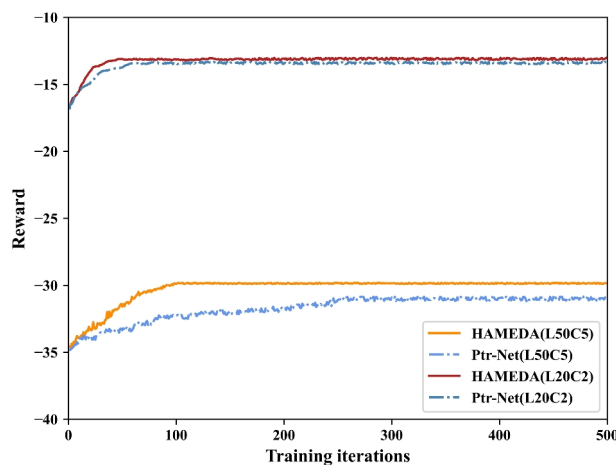


Fig. 5. Effect of training epochs on convergence and performance of HAMEDA model.

Discussion

The study presents a detailed description of the data generation methodology employed for the training and testing datasets, as well as the benchmark algorithms and hyperparameter settings utilized for the proposed HAMEDA model. The experimental setup involves random sampling of the coordinates of depots and LPs within a unit square defined by the interval $[0, 1] \times [0, 1]$, employing a uniform distribution. CSs are selected randomly from a discrete grid defined by $[0, 0.25, 0.5, 0.75, 1] \times [0, 0.25, 0.5, 0.75, 1]$. The weights of LPs are discrete values randomly chosen from the set $[1, 2, \dots, 6]$. It is assumed that the unit energy consumption and speeds remain constant before and after the loading of packages for each ELV. For the purpose of simplification, the vehicle speed is standardized for all ELVs to 1.0. To assess the performance of the proposed model, HAMEDA on instances of varying problem sizes are evaluated, specifically 20 LPs with 2 CSs, and 50 LPs with 5 CSs, which designated as L20C2 and L50C5, respectively. Furthermore, the performance of the proposed HAMEDA is compared against several representative baseline algorithms, which include:

1. *OPT*: The optimal solution to LECS problem is obtained using the standard Gurobi version 9.0.1 solver⁴⁵, which is recognized as a state-of-the-art exact solver for combinatorial optimization problems.
2. *Pointer Network (Ptr-Net)*: Ptr-Net is effective in addressing combinatorial optimization problems, utilizing an attention model to select a member from the input sequence as the output⁴⁶. The hyperparameters employed are consistent with those proposed in HAMEDA.
3. *Variable Neighborhood Search (VNS)*: An effective heuristic approach for addressing VRP and their variants⁴⁷.
4. *Kool-AM*: Kool-AM can leverage the graph attention mechanism to effectively capture the relationships between nodes, which is applied to solve combinatorial optimization problems. Following the idea of⁴⁸, we the model to the multi-agent scenario.

All algorithms were implemented using Python version 3.7.6. Furthermore, the implementations of Ptr-Net and HAMEDA were conducted utilizing PyTorch framework version 1.2.0 in conjunction with CUDA version 9.2. The experimental simulations were performed on a system running Ubuntu 16.04.6, which is equipped with a 32GB Tesla V100S GPU and an Intel® Xeon® Gold 5218R CPU with 500GB of storage. Each measurement was averaged over 1,000 iterations. A selection of parameters utilized for training and testing is presented in Table 2.

Performance results

This study initially conducted a comparative analysis of the reward training curves associated with the proposed HAMEDA algorithm across varying node counts to assess its convergence. In Figure 5, the reward function exhibits a marked increase corresponding to the rise in training iterations, indicating a continuous decrease in

the value of the objective function. Subsequently, HAMEDA algorithm demonstrates convergence, with L20C2 configuration achieving convergence in approximately 50 iterations, while L50C5 configuration converges in about 90 iterations. Notably, the convergence rate of HAMEDA surpasses that of Ptr-Net algorithm. Furthermore, HAMEDA ultimately attains a higher reward compared to Ptr-Net algorithm. In contrast to L20C2, L50C5 configurations of both algorithms exhibit a certain degree of fluctuation during the training iterations. However, HAMEDA algorithm maintains a relatively higher level of stability. This phenomenon can be attributed to the increasing number of LPs, which enhances the effectiveness of the constraints within the objective function. These observations indicate a significant improvement in the learning efficiency of HAMEDA algorithm, attributable to the implementation of the heterogeneous graph attention mechanism and the newly proposed decoding procedure utilized by HAMEDA network.

To validate the effectiveness of proposed HGA network, the one-agent-per-decoding-step strategy and masking process, Fig. 6 illustrates the reward changes over iterations for the HAMEDA algorithm, the HAMEDA without HGA, and the HAMEDA without improved decoding paradigm, i.e. the one-agent-per-decoding-step strategy and masking process, under scenarios with L50C5. The simulation results indicate that, compared to the HAMEDA algorithm without HGA, the proposed algorithm achieves higher rewards but with a slower convergence rate. This suggests that incorporating HGA network into the encoder allows agents to extract more hidden relationships among all nodes, leading the policy iteration closer to the optimal solution. The reason for the slow convergence speed is that the HGA network integrates self-attention and cross-attention mechanisms, extracts more features, leads to an increase in the dimensionality of the state and joint action space, and requires more iterations. Additionally, compared HAMEDA algorithm without improved decoding paradigm, the proposed algorithm demonstrates faster and more stable learning speeds and higher rewards. This is because the masking process effectively filters out infeasible nodes to reduce action space that proposed algorithm needs to learn and explore, thereby enhancing the algorithm's learning speeds and rewards. The one-agent-per-decoding-step strategy can enhance training stability. Thus, the HGA network, the one-agent-per-decoding-step strategy and masking process can effectively improve the performance of the HAMEDA algorithm.

The research subsequently analyzed the variations in total energy consumption across all algorithms with respect to different node configurations. The parameters are established as follows: a maximum travel time of 65, a maximum energy consumption of 15, and a maximum load constraint of 150. In order to intuitively show the impact of different numbers of LPs and CSs, we set different scales for the y-axis in Figure 7a and 7b. Figure 7a demonstrates that an increase in the number of LPs is correlated with a rise in the total energy consumption, as measured across all algorithms, with the number of CSs held constant at 5. This increase can be attributed to the fact that the total length of transportation routes typically expands with the increase in the number of users. Notably, the energy consumption reported by the proposed algorithm remains lower than that of the other algorithms. In Figure 7b, an increase in the number of CSs results in a decrease in the total energy consumption outputted by all algorithms, with the number of LPs maintained at 50. This reduction can be explained by the enhanced recharging strategy afforded by the additional CS nodes, which allows ELVs to extend their service range more effectively. Under the constraints of limited load and duration capacity, ELVs can cover a broader area, thereby facilitating the servicing of more LP nodes and significantly reducing overall system energy consumption. In comparison to Ptr-Net and Kool-AM, HAMEDA demonstrates superior performance in optimizing the objective function, attributable to the heterogeneous graph attention mechanism that adeptly extracts dynamic features and relationships among all nodes, including both CS and ELV nodes.

This study conducted an analysis of the effect of varying constraint values on total energy consumption in Figure 8 with the number of LPs and CSs set at 50 and 5, respectively. The results indicate that as the values of several constraints, including maximum transportation duration, maximum battery capacity, and maximum load capacity, increase, the total energy consumption of the different algorithms tends to decrease. This reduction in energy consumption among all ELVs can be ascribed to higher constraint settings, which enable the vehicles to service an increased number of LP nodes.

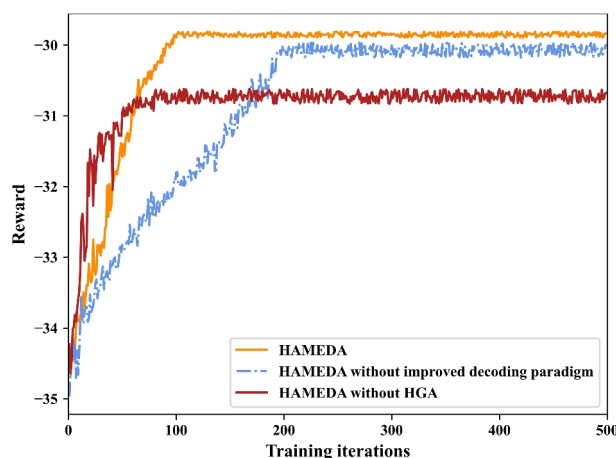


Fig. 6. The effect of HGA and improved decoder.

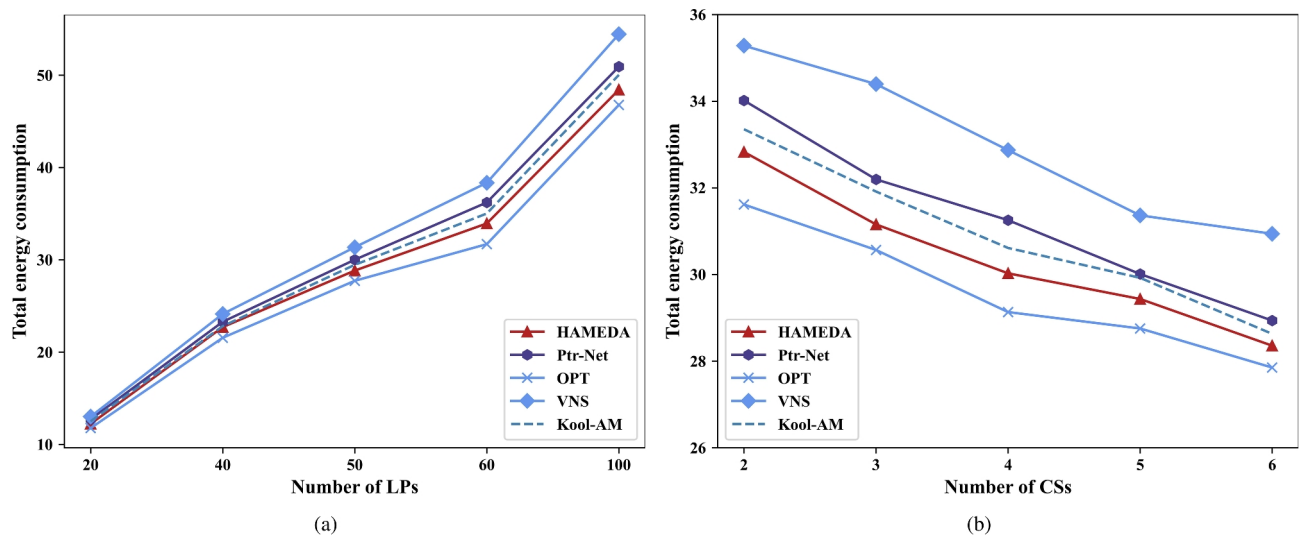


Fig. 7. Effects of LPs and CSs on total energy consumption: (a) Total energy consumption versus LPs; (b) Total energy consumption versus CSs.

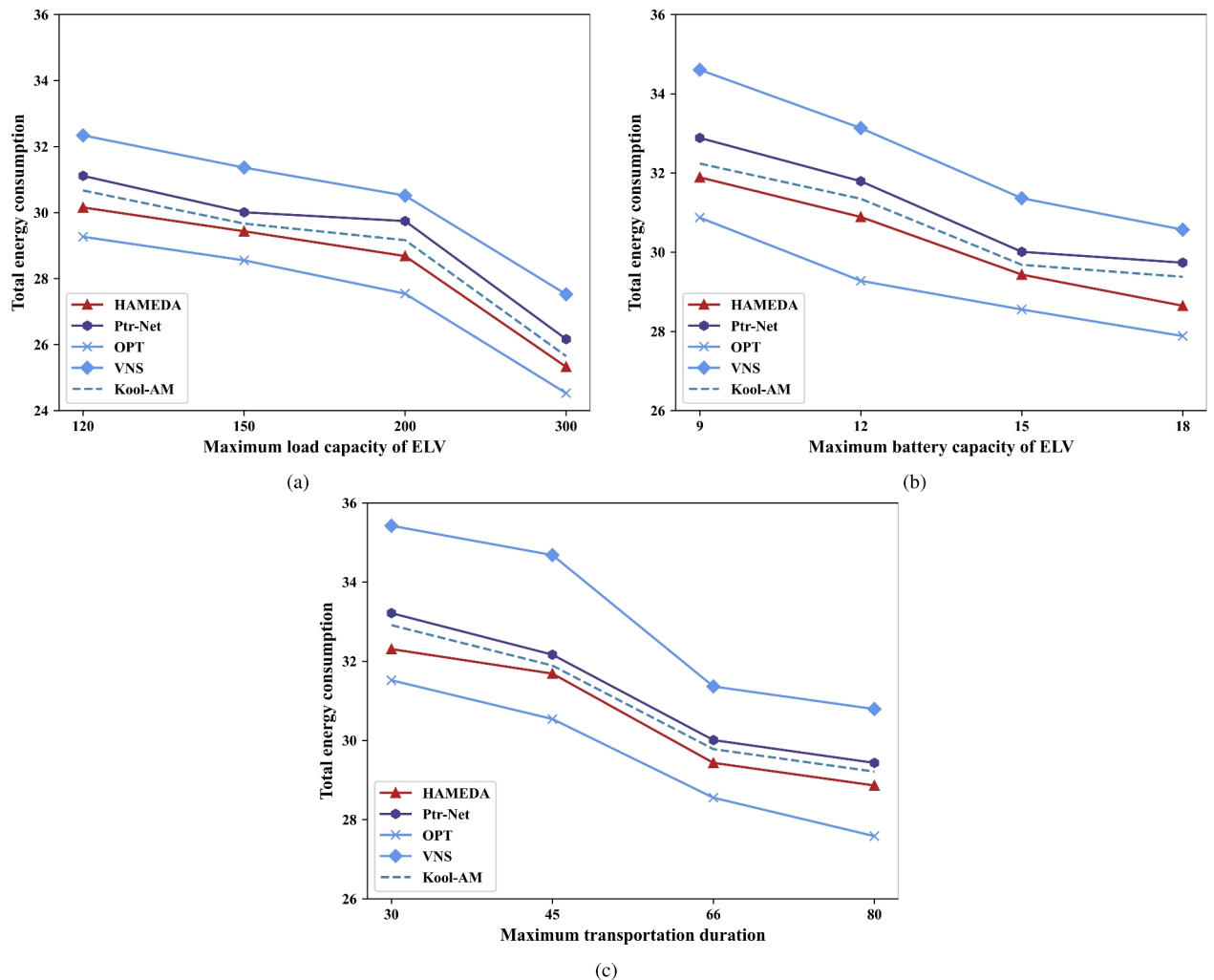


Fig. 8. Effects of different constraints on total energy consumption in ELVs: (a) Maximum load capacity; (b) Maximum battery capacity; (c) Maximum transportation duration.

Algorithms	L20C2	L50C5	L100C10
OPT	37.40 s	4479 s	6075 s
Ptr-Net	0.87 s	1.08 s	1.29 s
VNS	21.16 s	416.38 s	860.57 s
Kool-AM	0.92 s	1.17 s	1.86 s
HAMEDA	0.91 s	1.03 s	1.34 s

Table 3. Comparative running times of OPT, HAMEDA, Ptr-Net and VNS.

Table 3 details the execution times of the OPT, HAMEDA, VNS, Ptr-Net and Kool-AM algorithms, noting that the duration required by each algorithm extends as the node count rises. Notably, the runtime of OPT and VNS algorithms exhibits an almost exponential growth as the problem size escalates, whereas the runtime of the learning-based methods demonstrates a linear growth pattern. Specifically, the runtimes of OPT and VNS methods are significantly longer than those of the learning-based approaches, which consistently remain approximately 1s. Furthermore, HAMEDA consistently yields a solution that is comparable in quality while requiring relatively less execution time for large-scale instances.

Comparatively, the HAMEDA algorithm consistently demonstrates at least a 1.64% lower total energy consumption than other DRL-based algorithms, i.e. Ptr-Net and Kool-AM, yet it shows 3.5% higher when compared to OPT. Moreover, HAMEDA significantly outperforms other baseline algorithms in execution speed for large-scale scenarios, thus enhancing timely decision-making capabilities. The experimental findings underscore HAMEDA’s ability to strike an effective balance between quick execution and high-quality solutions, alongside marked improvements in optimization efficacy and system scalability.

Conclusion

This paper addresses LECS problem in urban logistics, where multiple CSs are available for recharging ELVs. LECS problem is formulated to minimize the total energy consumption of heterogeneous ELVs, each with different load capacities and battery sizes, under the constraint of maximum transportation duration. Given the NP-hard nature of LECS problem, it was modeled as an MDP and proposed HAMEDA approach, based on an Encoder-Decoder architecture. HAMEDA incorporates a heterogeneous graph attention network to capture advanced representations of the relationships between LPs and CSs during the encoding phase. In the decoding phase, a novel procedure that utilizes a one-agent-per-decoding-step routine and a masking strategy is introduced to enhance solution quality in tour generation, thereby improving learning efficiency. HAMEDA model is trained through DRL in an unsupervised manner. Extensive experiments demonstrate that HAMEDA reduces total energy consumption by at least 1.64% compared to traditional heuristic and learning-based algorithms. Furthermore, HAMEDA consistently delivers comparable solutions with significantly reduced execution times, making it an optimal choice for large-scale tasks that require rapid decision-making.

Our current research focuses on low energy consumption scheduling in scenarios with fixed CSs. However, in regions where fixed CSs are sparsely deployed or nonexistent, the coordinated scheduling of mobile CSs and ELVs remains an open challenge for future exploration. Additionally, the proposed centralized approach relies on the logistics cloud platform for all computations, potentially leading to significant data exchange between ELVs and the cloud. To further enhance efficiency and scalability, future research will explore advanced distributed methodologies, such as decentralized federated learning⁴⁹, to reduce communication overhead and improve overall system performance.

Data availability

The datasets used, generated and analyzed during this study are available from the corresponding author on reasonable request.

Received: 19 December 2024; Accepted: 3 March 2025
Published online: 15 March 2025

References

1. Greenpeace. Carbon emissions report of china’s express delivery industry by greenpeace organization. <https://www.greenpeace.org.cn> (2023).
2. Mou, J. et al. A machine learning approach for energy-efficient intelligent transportation scheduling problem in a real-world dynamic circumstances. *IEEE Trans. Intell. Transp. Syst.* **24**, 15527–15539 (2023).
3. Zhu, J., He, C., Cheung, K., Luo, F. & Liu, Y. Low carbon planning of multiple integrated energy systems considering trans-regional battery logistics network. *IEEE Trans. Sustain. Energ.* **15**, 1239–1255 (2024).
4. Sun, P., Li, L. & Wan, J. LPPCM: A low-cost package pickup covering mechanism for cooperative express services. *IEEE Trans. Sustainable Comput.* **9**, 386–395 (2024).
5. Zhang, L. et al. Routing optimization of shared autonomous electric vehicles under uncertain travel time and uncertain service time. *Transp. Res. Part E: Logist. Transp. Rev.* **175**, 102548 (2022).
6. Jiang, L. et al. Cooperative package assignment for heterogeneous express stations. *IEEE Trans. Intell. Transp. Syst.* **23**, 8467–8476 (2021).
7. Xiang, C., Wu, Z., Tu, J. & Huang, J. Centralized deep reinforcement learning method for dynamic multi-vehicle pickup and delivery problem with crowdshippers. *IEEE Trans. Intell. Transp. Syst.* **25**, 9253–9267 (2024).
8. Yao, C., Chen, S. & Yang, Z. Joint routing and charging problem of multiple electric vehicles: A fast optimization algorithm. *IEEE Trans. Intell. Transp. Syst.* **23**, 8184–8193 (2022).

9. Zhang, L.-Y., Tseng, M.-L., Wang, C.-H., Xiao, C. & Fei, T. Low-carbon cold chain logistics using ribonucleic acid-ant colony optimization algorithm. *J. Cleaner Prod.* **233**, 169–180 (2019).
10. Zhang, S. et al. A low-carbon, fixed-tour scheduling problem with time windows in a time-dependent traffic environment. *Int. J. Prod. Res.* **61**, 6177–6196 (2023).
11. Nan, Y. Multiobjective optimization for vehicle routing optimization problem in low-carbon intelligent transportation. *IEEE Trans. Intell. Transp. Syst.* **24**, 13161–13170 (2023).
12. Li, J. et al. Deep reinforcement learning for solving the heterogeneous capacitated vehicle routing problem. *IEEE Trans. Cybern.* **52**, 13572–13585 (2022).
13. Liu, B., Ni, W., Liu, R. P., Guo, Y. J. & Zhu, H. Optimal electric vehicle charging strategies for long-distance driving. *IEEE Trans. on Veh. Technol.* **73**, 4949–4960 (2024).
14. J., W. R. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* **8**, 229–256 (1998).
15. Wang, J., Guo, Q. & Sun, H. Planning approach for integrating charging stations and renewable energy sources in low-carbon logistics delivery. *Appl. Energy* **372**, 123792 (2024).
16. Qiang, H., Ou, R., Hu, Y., Wu, Z. & Zhang, X. Path planning of an electric vehicle for logistics distribution considering carbon emissions and green power trading. *Sustainability* **15**, 16045 (2023).
17. Liang, Z., ho Chiu, Y., Li, X., Guo, Q. & Yun, Y. Study on the effect of environmental regulation on the green total factor productivity of logistics industry from the perspective of low carbon. *Sustainability* **12**, 175 (2019).
18. Jiang, J., Zhang, D., Meng, Q. & Liu, Y. Regional multimodal logistics network design considering demand uncertainty and co2 emission reduction target: A system-optimization approach. *J. Cleaner Prod.* **248**, 119304 (2020).
19. Zhang, S., Chen, N., Song, X. & Yang, J. Optimizing decision-making of regional cold chain logistics system in view of low-carbon economy. *Transp. Res. Part A: Policy Pract.* **130**, 844–857 (2019).
20. Ren, Y., Wang, C., Li, B., Yu, C. & Zhang, S. A genetic algorithm for fuzzy random and low-carbon integrated forward/reverse logistics network design. *Neural Comput. Appl.* **32**, 2005–2025 (2020).
21. Zhang, D., Zhan, Q., Chen, Y. & Li, S. Joint optimization of logistics infrastructure investments and subsidies in a regional logistics network with co2 emission reduction targets. *Transp. Res. Part D: Transp. Environ.* **60**, 174–190 (2018).
22. Wang, J., Lim, M. K., Tseng, M.-L. & Yang, Y. Promoting low carbon agenda in the urban logistics network distribution system. *J. Cleaner Prod.* **211**, 146–160 (2019).
23. Lim, Y. L. M. K., Tan, Y., Lee, S. Y. & Tseng, M.-L. Sharing economy to improve routing for urban logistics distribution using electric vehicles. *Resour. Conserv. Recycl.* **153**, 104585 (2020).
24. Zhang, X., Jin, F.-Y., Yuan, X.-M. & Zhang, H.-Y. Low-carbon multimodal transportation path optimization under dual uncertainty of demand and time. *Sustainability* **13**, 8180 (2021).
25. Wu, T., Li, Z., Wang, G., Zhang, X. & Qiu, J. Low-carbon charging facilities planning for electric vehicles based on a novel travel route choice model. *IEEE Trans. Intell. Transp. Syst.* **24**, 5908–5922 (2023).
26. Zhang, H., Huang, Q., Ma, L. & Zhang, Z. Sparrow search algorithm with adaptive t distribution for multi-objective low-carbon multimodal transportation planning problem with fuzzy demand and fuzzy time. *Expert Syst. Appl.* **238**, 122042 (2024).
27. A., K. M. Transportation cost optimization using linear programming. in *International conference on mechanical, industrial and energy engineering, MEIE2014*, 2241–2245 (2014).
28. Wu, S. et al. Cooperative scheduling for directional wireless charging with spatial occupation. *IEEE Trans. Mobile Comput.* **23**, 286–301 (2022).
29. Ou, J. et al. Solving many-objective delivery and pickup vehicle routing problem with time windows with a constrained evolutionary optimization algorithm. *Expert Syst. Appl.* **255**, 124712 (2024).
30. Li, K. et al. Deep reinforcement learning for combinatorial optimization: Covering salesman problems. *IEEE Trans. Cybern.* **52**, 13142–13155 (2022).
31. Fan, M. et al. Deep reinforcement learning for uav routing in the presence of multiple charging stations. *IEEE Trans. on Veh. Technol.* **72**, 5732–5746 (2023).
32. Zhang, R. et al. Learning to solve multiple-tsp with time window and rejections via deep reinforcement learning. *IEEE Trans. Intell. Transp. Syst.* **24**, 1325–1336 (2023).
33. Ruiz, J., Gonzalez, C., Chen, Y. & Tang, B. Prize-collecting traveling salesman problem: A reinforcement learning approach. *IEEE International Conference on Communications* **2023**, 4416–4421 (2023).
34. Zhao, J., Mao, M., Zhao, X. & Zou, J. A hybrid of deep reinforcement learning and local search for the vehicle routing problems. *IEEE Trans. Intell. Transp. Syst.* **22**, 7208–7218 (2021).
35. He, Y., Liu, Y., Yang, L. & Qu, X. Deep adaptive control: Deep reinforcement learning-based adaptive vehicle trajectory control algorithms for different risk levels. *IEEE Trans. Intell. Veh.* **9**, 1654–1666 (2024).
36. Zhang, C., Liu, Y., Wu, F., Tang, B. & Fan, W. Effective charging planning based on deep reinforcement learning for electric vehicles. *IEEE Trans. Intell. Transp. Syst.* **22**, 542–554 (2021).
37. Deppon. Price rules of deppon express. <https://www.deppon.com/> (2024).
38. Bard, J. F. A branch and cut algorithm for the vrp with satellite facilities. *IIE Trans.* **30**, 821–834 (1998).
39. Erdogan, S. & Miller-Hooks, E. A green vehicle routing problem. *Transp. Res. Part E: Logist. Transp. Rev.* **48**, 100–114 (2012).
40. Peng, B., Wang, J. & Zhang, Z. A deep reinforcement learning algorithm using dynamic attention model for vehicle routing problems. *Artificial Intelligence Algorithms and Applications: 11th International Symposium, ISICA 2019*, 636–650 (2019).
41. Xu, Y. et al. Bnet: Batch normalization with enhanced linear transformation. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**, 9225–9232 (2023).
42. A., V. Attention is all you need. in *Proc. Adv. Neural Inf. Process. Syst.* **1**, 30 (2017).
43. Du, Y., Pei, B., Zhao, X. & Ji, J. Deep scaled dot-product attention based domain adaptation model for biomedical question answering. *Methods* **173**, 69–74 (2020).
44. Kingma, D. P. Adam: A method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980), (2014).
45. Optimization, L. G. Gurobi optimizer reference manual. <https://www.gurobi.com/products/gurobi-optimizer/> (2020).
46. Veličković, P. et al. Pointer graph networks. *Adv. Neural Inf. Process. Syst.* **33**, 2232–2244 (2020).
47. Xu, Z. & Cai, Y. Variable neighborhood search for consistent vehicle routing problem. *Expert Syst. Appl.* **113**, 66–76 (2018).
48. Sankaran, P., McConky, K., Sudit, M. & Ortiz-Peña, H. Gamma: Graph attention model for multiple agents to solve team orienteering problem with multiple depots. *IEEE Trans. Neural Netw. Learn. Syst.* **34**, 9412–9423 (2023).
49. Yuan, L. et al. Decentralized federated learning: A survey and perspective. *IEEE Internet Things J.* **11**, 34617–34638 (2024).

Acknowledgements

This research is supported in part by the National Natural Science Foundation of China under grants 62362055, in part by the Natural Science Foundation of Inner Mongolia Autonomous Region under grants 2024MS06021 and 2024QN06014, and in part by the Research Projects of Universities in Inner Mongolia Autonomous Region (JY20230058).

Author contributions

P.S. conceived the conceptualization and methodology, J.H., J.W. and Y.G. conducted the experiments, D.L. and X.S. analysed the results, L.L. reviewed and edited manuscript. All authors have read and agreed to the published version of the manuscript.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to P.S. or L.L.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025