Received 1 May 2024, accepted 8 May 2024, date of publication 10 May 2024, date of current version 18 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3399610

RESEARCH ARTICLE

Real-Time Multi-Vehicle Scheduling in Tasks With Dependency Relationships Using Multi-Agent Reinforcement Learning

SHUPEI ZHANG, HUAPENG SHI, WEI ZHANG, YING PANG, AND PENGJU SUN
School of Automotive and Traffic Engineering, Jiangsu University, Zhenjiang 212013, China

Corresponding author: Huapeng Shi (1085790087@qq.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 52272366.

ABSTRACT With the advancement of technology in vehicle-road collaboration and autonomous driving, new commercial applications have surfaced. These include autonomous ride-hailing vehicles and unmanned delivery vehicles. As a result of the challenges presented by commercial applications, dispatching systems are moving towards being maintenance-free, centralized, multitasking, and real-time. Yet, most existing dispatching systems have been designed for single-task purposes and cannot tackle multitasking issues. Moreover, traditional optimization algorithms make it difficult to achieve timeliness in real-time changing traffic conditions. Therefore, this paper innovatively proposes a task allocation method based on Multi-Agent Reinforcement Learning (MARL). Firstly, this study introduces a classification model of task relationships through the binary assumption model of geographical areas and vehicles. Secondly, the study matches the classification model's task cost state transition process with the Markov Decision Process, constructing a Multi-Agent Reinforcement Learning framework. Finally, the study constructs a simulation environment suitable for reinforcement learning based on Simulation of Urban Mobility (SUMO). Simulation results indicate that the task allocation system based on MARL can effectively improve the system's overall efficiency by determining the order of task allocation and the matching relationships between tasks.

INDEX TERMS Combinatorial optimization, multi-agent system, reinforcement learning, simulation of urban mobility, task allocation.

I. INTRODUCTION

The communication information between vehicles, as well as between cars and operational infrastructure, can be further transmitted to the cloud system, forming the basic structure of vehicle road collaboration [1]. Based on this information, the cloud decision-making system can allocate tasks to each vehicle reasonably, ensuring that each vehicle can fully utilize its capabilities and complete as many tasks as possible [2]. Ren [3] believes that reasonable task allocation and path optimization can significantly reduce operational costs. At the same time, Yu, J. [4] also proves that by focusing on system costs, total driving distance, and delay indicators, the system's overall efficiency can be improved as much as possible.

Nowadays, advancements in task allocation systems have complemented the development of unmanned driving technology [5]. This technology enables the coordination of several unmanned vehicles to accomplish tasks as a team. At container terminals, automated-guided vehicles (AGVs) are navigated to designated areas to coordinate the handling of containers [6]. In the ride-hailing industry, autonomous vehicles must coordinate with each other to minimize passenger wait times [7]. In underground mines, due to narrow roadways, multiple unmanned mining trucks need to coordinate their routes to maximize transport capacity [8]. In the express delivery industry, autonomous delivery robots must collaborate to reduce delivery expenses and maximize the number of fulfilled orders [9].

The associate editor coordinating the review of this manuscript and approving it for publication was Qiang Li.

However, the introduction of autonomous driving technology has raised higher requirements for task allocation technology. Dispatching systems should evolve to become maintenance-free and capable of multitasking coordination [10]. For instance, in the past, taxi drivers would personally take care of their vehicles by charging and cleaning them after each passenger trip. However, autonomous ride-hailing companies need a more efficient dispatch system that can allocate tasks for each vehicle uniformly. These tasks include transporting passengers, charging, and cleaning.

Furthermore, these tasks are not completely independent of each other. Multiple tasks can interact and become constraints for each other. If a vehicle with a low battery is assigned a task that requires it to travel far from a charging station, it will take longer for the vehicle to reach the charging station. In such scenarios, the scheduling system should shift from single-objective optimization to multi-objective optimization to coordinate multiple tasks effectively.

However, considering the interdependence of various tasks and the need to set corresponding constraints, task assignment becomes more complex, significantly increasing the time required for the solution. Finding the optimal strategy for task assignment has been proven to be a challenging problem [9]. It is challenging to balance accuracy and speed when allocating multiple objectives, especially in vehicle scheduling. This is because tasks arise in real time, and the traffic environment frequently changes. As a result, scheduling systems need to make quick decisions.

For the analysis above, the overall objective of this paper is to design a vehicle task assignment system that can coordinate multiple tasks and have real-time capabilities. Many current vehicle scheduling systems are designed to optimize for multiple objectives in static scenarios where tasks do not change in real-time [11]. Others use graph theory or reinforcement learning for real-time optimization of a single objective [12]. However, there is currently no established system architecture capable of scheduling vehicles based on multiple objectives in dynamic scenarios. This research aims to introduce new approaches to the field of multi-vehicle dispatching. The main contributions of this study are summarized below:

1. This study creates a classification model that can be used in vehicle tasks and describes the interdependent relationship between multiple tasks. The model assumes that each task completed by a vehicle incurs a cost and can analyze the process of cost transition. This analysis will provide a theoretical framework for constructing a reinforcement learning model.
2. This study presents a framework that uses multi-agent reinforcement learning to assign tasks to vehicles. In this architecture, agents guide the vehicles to pick up tasks through a policy network. As each vehicle completing a task incurs a corresponding cost, the state value network always represents the total estimated cost of remaining tasks. This means that the value network continuously updates the agent's policy network towards lower costs.
3. A simulation system has been developed based on Simulation of Urban Mobility (SUMO), which is suitable for training reinforcement learning. The system provides a considerable amount of data and enables real-time interaction for the intelligent agent during its training. Furthermore, the simulation system supports the replacement of maps, vehicle tasks, and basic models of reinforcement learning based on the architecture of the study.

The basic structure of this study is shown in Figure 1, and the rest of this article is organized as follows.

Section II summarizes the theoretical framework and research status of Multi-Robot Task Allocation and explains reinforcement learning and its application in task assignments. Section III proposes mathematical models and optimization objectives to analyze the dependencies in vehicle tasks. Section IV proposes the principles and construction methods of multi-agent reinforcement learning architecture. Section V compares the simulation results based on the simulator. At last, Section VI concludes the paper.

II. BACKGROUND

This section briefly introduces the relevant theories and current research status of Multi-Robot Task Allocation (MRTA) and explains the reinforcement learning methods and their application in the field of task allocation.

A. OVERVIEW OF MRTA

The research on multi-robot systems (MRS) [13] focuses on designing and controlling multiple robots to collaborate and complete tasks. Multi-Robot Task Allocation(MRTA) is a critical problem in MRS [14], which can be traced back to classic operations research problems such as the Traveling Salesman Problem (TSP) and Vehicle Routing Problem (VRP). However, unlike these traditional problems, Multi-Robot Task Allocation typically includes time constraints and the need to consider dynamic and uncertain environments. With the popularization of unmanned aerial vehicles, autonomous vehicles, and other autonomous systems, the importance of MRTA has further increased, and its application areas have gradually expanded to include search and rescue [15], agricultural monitoring [16], and logistics distribution [17].

Currently, many task allocation algorithms for multi-agent systems have been proposed by researchers. These algorithms can be categorized as follows: distributed full search algorithms [18], distributed local search algorithms [19], auction-based mechanisms [20], distributed particle swarm optimization [21], and distributed ant colony algorithms [22].

However, the algorithms mentioned in this study still have some shortcomings when it comes to dynamic multi-vehicle scenarios. Distributed full search algorithms can provide the optimal solution for task allocation, but they come with high communication and computational costs. Some scholars have developed event-triggered multi-agent control algorithms to reduce the dependence on global information [23], [24], but

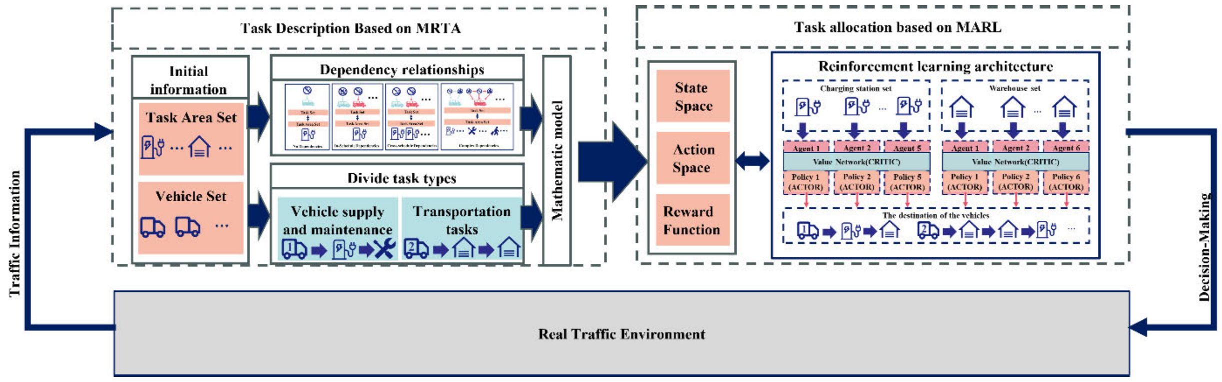


FIGURE 1. Abstract view of novel contribution.

it is still unrealistic to use them when scheduling a large number of vehicles. Distributed local search algorithms cannot guarantee the quality or diversity of solutions. Auction mechanisms rely on artificially designed evaluation criteria and cannot optimize multiple objectives simultaneously. Lastly, heuristic algorithms still lack systematic research and evaluation in dynamic scenarios [25].

This research involves developing algorithms capable of managing dynamic scenarios involving multiple vehicles while also taking into consideration the constraint relationships among different tasks within these vehicles. Several scholars have suggested classification methods for these constraint relationships [26], [27], [28]. This study has adopted the iTAX method [29] as the theoretical foundation, which can describe both the instantaneous, independent relationships among similar tasks and the extended, interrelated multi-step relationships among different tasks. The approach identifies four main types of task relationships, which are illustrated in Figure 2.

1. No Dependencies (ND): The cost of a given pair robot task is independent of all others.
2. In-Schedule Dependencies (ID): The cost of a given pair robot-task depends on the other tasks assigned to this robot. It is intra-schedule dependencies.
3. Cross-Schedule Dependencies (XD): The cost of a given pair robot-task depends not only on the other tasks assigned to this robot but also on other robots' schedules. It is inter-schedule dependencies.
4. Complex Dependencies (CD): Under this dependency, heterogeneous tasks are involved. On the basis of cross-scheduling dependencies, the cost of a given pair of robot tasks also depends on the constraint relationship between heterogeneous tasks. Therefore, the task decomposition problem is coupled with the task allocation problem, and individual agents cannot be decoupled from each other.

B. OVERVIEW OF RL

Reinforcement learning has certain advantages over traditional optimization methods when it comes to solving dynamic problems. Compared with optimization methods that require iterative optimization to find solutions, reinforcement learning involves training a neural network to forward propagate the relevant parameters of the solution. The solving speed is only related to the complexity of the network [30]. Therefore, reinforcement learning has been used as a solution for real-time problems in many fields [31], [32], [33].

In reinforcement learning, problems are typically modeled as Markov Decision Processes (MDP), where the decision-maker selects one action from a set of possible actions at each time step to interact with their environment and receives a reward and a new state from the environment. Under this framework, the Bellman equation plays a central role, outlining the relationship between the state-value function or action-value function and their subsequent states or actions. These equations provide a theoretical foundation for finding the optimal strategy, with the action-value function's Bellman equation being as follows:

$$Q^{\pi}(s,a) = \sum_{s',r} p\left(s',r \mid s,a\right)\left[r+\gamma \sum_{a}^{\prime} \pi\left(a' \mid s'\right) Q^{\pi}\left(s',a'\right)\right] \quad (1)$$

where $Q^{\pi}(s,a)$ is the expected return of taking action a in state s and following policy π; s, s' are the current and next states; a, a' are the current and next actions; $p\left(s',r \mid s,a\right)$ is the probability of transitioning from state s taking action a to state s' and receiving reward r; γ is the discount factor; $\pi\left(a' \mid s'\right)$ is the probability of choosing action a' in state s' according to policy π. The state-value Bellman equation is:

$$V^{\pi}(s) = \sum_{a} \pi(a \mid s) \sum_{s',r} p\left(s',r \mid s,a\right)\left[r+\gamma V^{\pi}\left(s'\right)\right] \quad (2)$$

where $V^{\pi}(s)$ is the expected return of following policy π in state s; s, s' are the current and next states; a is the action taken in state s; $\pi(a \mid s)$ is the probability of taking action a in state s; $p\left(s',r \mid s,a\right)$ is the probability of transitioning from state s taking action a to state s' and receiving reward r; γ is the discount factor.

Based on the two value functions mentioned above, reinforcement learning enables a single agent to learn the optimal

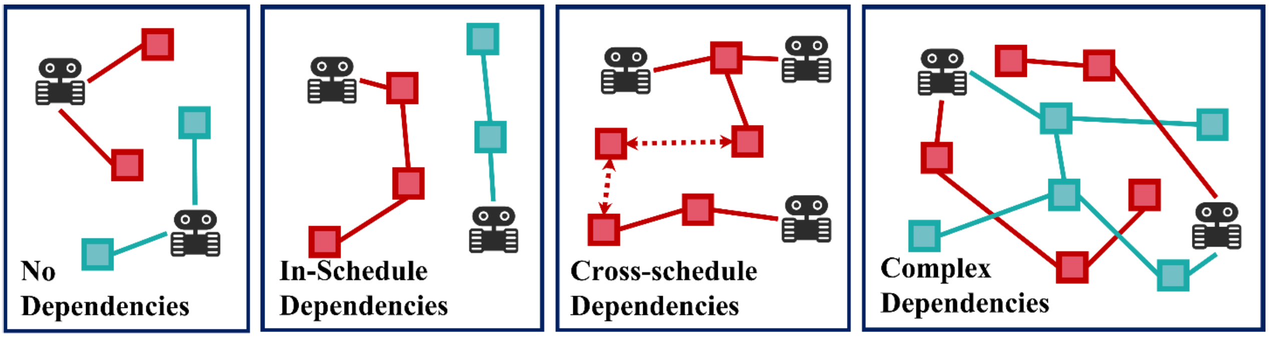


FIGURE 2. Four dependencies in the iTAX classification model.

strategy through two fundamental learning methods: value learning [34] and policy learning [35]. Value learning focuses on estimating the value of taking specific actions in a given state, while policy learning directly learns a policy that maps states to action probability distributions. By employing these two methods, intelligent agents can gradually enhance their decision-making quality in complex and uncertain environments.

However, constructing a reinforcement learning framework for multi-task allocation remains a challenge. Existing reinforcement learning methods are usually applied to multi-task allocation in unmanned aerial vehicles [36] or multiple robots [37]. In these fields, the number of agents involved is usually small. This makes it easier for the reinforcement learning value network to converge during training. However, in the field of vehicle scheduling, there are usually a significantly greater number of vehicles involved. This leads to a large number of corresponding agents, which makes it difficult for the reinforcement learning value network to converge during training [38]. As a result, the existing multi-agent reinforcement learning frameworks lack generalizability in vehicle task allocation problems.

In the field of vehicle dispatching, reinforcement learning is not directly used for vehicle allocation tasks. Instead, it is applied to balance the demand for vehicle usage, indirectly affecting vehicle allocation. Song et al. utilize Deep Reinforcement Learning (DRL) to determine the demand for ride-sharing vehicles in different areas at different times to reduce waiting times for vehicles [39]. Liu et al. propose transforming the action space of Deep Reinforcement Learning into a recommendation list to address high-concurrency dispatch requests [40], guiding taxis to passenger areas. In dealing with the road network in taxi dispatching issues, Liu et al. introduce a new network segmentation method to allow deep neural networks to better contextual awareness [41], thereby reducing passenger waiting times. Jiao et al. Utilizing reinforcement learning [42] to enhance the economic benefits of addressing the vehicle repositioning issue on ride-hailing platforms. These methods measure the car demand in different areas through reinforcement learning, indirectly dispatching vehicles, thus making it possible to dispatch multiple vehicles. However, these methods only consider single-task requirements and are not suitable for use in the multi-vehicle, multi-task scenario.

A system composed of multiple agents is referred to as multi-agent reinforcement learning (MARL) in the field of reinforcement learning, where each agent tends to use policy-based learning. In a multi-agent environment, each agent's actions have an impact on the others, thus affecting their state and return. This interdependence leads to a dynamic environment for individual agents, where changes in other agents' strategies drive the dynamics. For this reason, policy learning methods are advantageous in adapting, as they enable agents to respond more directly to changes in other agents' behaviors.

The Actor-Critic method, which integrates the advantages of policy gradient (Actor) and value function (Critic), achieves an effective balance of policy and value learning through its dual-component structure [43]. The Actor, serving as the policy function, often uses a parameterized form (such as neural networks) to map a given state to a probability distribution of actions, aiming to learn a policy that maximizes long-term cumulative rewards. On the other hand, the Critic is responsible for evaluating the effects of actions chosen by the Actor, using a parameterized value function to estimate the value of a given state or state-action pair. The interaction of these two components is key to the Actor-Critic method.

Based on the above analysis, this study adopts a fully cooperative multi-agent reinforcement learning framework and uses the multi-agent Advantage Actor-Critic (A2C) method [44]. In this setup, all agents share the same goal to maximize the overall performance of the system. Its schematic is shown in Figure 3, where all agents have both a value network v and a policy network π, sharing a common value network and having their own policy networks. In a fully cooperative context, the value network helps agents evaluate the contribution of their actions to the overall goal. The policy network is responsible for generating specific action strategies.

III. ANALYTICAL MODEL

This section will simplify all tasks into a binary model of region-vehicle, based on an analysis of the vehicle dispatching task. Based on the binary model, this section

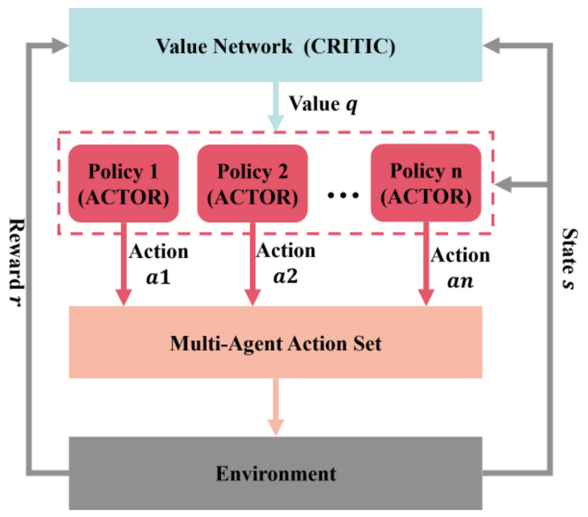


FIGURE 3. Multi-agent actor-critic methods.

proposes a task categorization model using the iTAX model. Furthermore, this section will establish the corresponding mathematical model based on the simplified binary model and analyze the optimization objective under this mathematical model. i

A. VEHICLE TASK CLASSIFICATION MODEL

With the application of unmanned driving technology in the commercial field, the vehicle dispatching system is gradually developing in the direction of multi-task collaboration, so the analysis of the dependency between vehicle tasks is conducive to the design of the dispatching system. In terms of commercial categories, vehicle tasks encompass a wide range, such as "attended home delivery", "same-day delivery", and "mobility-on-demand". However, from the actual subject of the task, where tasks can be broadly categorized into transportation tasks [45] and vehicle supply and maintenance tasks [46]. The information corresponding to these two types of tasks is shown in Table 1. Transportation tasks are typically requested by passengers or for moving goods. These tasks are dependent on the locations that need to be reached and are constrained by the carrying capacity of the vehicles. It is possible that a single transportation task may require the use of multiple vehicles to complete. Vehicle supply and maintenance tasks can often limit transportation options. These tasks are typically required by the vehicles themselves, as they require regular maintenance and charging. Unlike transportation tasks, vehicle maintenance can be performed anywhere, but the corresponding vehicles must be available for maintenance.

Due to the divergent nature of the two types of task attributes, scheduling systems are not conducive to analyzing tasks. Therefore, this study makes the following two assumptions.

1. All task requirements are considered from the perspective of vehicles and correspond to TASK in Multi-Robot Task Allocation. In order to standardize the framework, this paper considers all tasks from the perspective of vehicles, collectively referred to as vehicle requirements. From a certain perspective, the transportation system itself exists to meet the needs of various vehicles, such as fuel or energy supply requirements, road passage requirements, passenger-carrying requirements, parking requirements, etc. These different requirements correspond to different tasks. The transportation task can also be seen as vehicles traveling on the corresponding road sections to meet operational needs.
2. Correspond traffic zones and infrastructure in Multi-Robot Task Allocation as ROBOT. The diverse facilities and road networks in the transportation system function similarly to robots in Multi-Robot Task Allocation systems. Taking ride-hailing services as an example, the varying passenger flow in different areas at different times provides vehicles with different passenger-carrying opportunities. At the same time, specific areas are equipped with charging facilities to meet the vehicles' endurance needs. The design and layout of transportation facilities directly affect the efficiency and functionality of the transportation system. Additionally, transportation facilities also have resource constraints, such as the traffic flow volume allowed at intersections, the number of charging vehicles that charging stations can support, and the vehicle capacity that parking lots can accommodate. These resource constraints require vehicle scheduling systems to have better resource allocation capabilities.

TABLE 1. Task attributes in transportation systems.

	Transportation	Vehicle supply and maintenance
Number of waypoints	≥ 2	1
Dependencies among waypoints or not	Yes	No
Number of required vehicles	≥ 1	1
Designated waypoint or not	Yes	No
Designated vehicle or not	No	Yes
Instance	Passenger transport, Freight transport	Battery charging, Parking

It should be noted that in the context of Multi-Robot Task Allocation, ROBOT is typically a movable object. However, in the assumption of this study, it is the transportation facilities that complete the tasks, and they do not possess the ability to move on their own. Vehicles need to move to them instead. However, from the perspective of topology, there is no distinction in terms of who serves as the moving subject. As long as the travel time of the moving subject is taken into account by the task assignment system. However, this assumption fixes the number of agents and is advantageous for the construction of multi-agent reinforcement learning [47].

After simplifying all vehicle tasks into a binary relationship of region-vehicle, it is possible to analyze the dependency relationships within vehicle tasks based on the iTAX model and derive the following four classifications, as depicted in Figure 4.

1. No Dependencies in vehicle tasks: The cost of a given pair vehicle-task is independent of all others, meaning there is only one vehicle at a time, and the vehicle only needs to pass through waypoints in a fixed order.
2. In-Schedule Dependencies in vehicle tasks: The cost of a given pair vehicle-task depends on the other tasks assigned to this task area. In this case, the same task area faces multiple vehicles, so the task completion progress depends on the vehicle arrival sequence.
3. Cross-Schedule Dependencies in vehicle tasks: The cost of a given pair vehicle-task depends not only on the other tasks assigned to this task area but also on other task areas' schedules. In this case, there will be multiple task areas and corresponding multiple vehicles. Therefore, matching too many vehicles in a particular task area can result in other task areas being unable to obtain vehicles, thereby delaying the progress of tasks in other areas.
4. Complex Dependencies in vehicle tasks: Based on the cross-schedule dependencies, this dependency relationship involves multiple categories of tasks. These different types of tasks will serve as constraints for each other. The cost of a given pair of vehicle tasks depends on the order in which different types of tasks are completed. Therefore, the vehicle task decomposition problem is coupled with the vehicle task assignment problem.

Assuming that there is a cost associated with each vehicle completing each task, based on the above four categories, the vehicle scheduling system can better analyze the sources of cost in practical tasks. Particularly in reinforcement learning, both the reward function and state value are related to this cost. Therefore, the state space transition process of reinforcement learning needs to be matched with the cost state transition process. In section IV, this study will analyze in detail how to design a corresponding reinforcement learning framework based on dependencies.

B. MATHEMATICAL MODEL AND OPTIMIZATION OBJECTIVE

Building on the binary hypothesis of vehicle and task area mentioned in the previous section, the theoretical framework mentioned has characterized all traffic tasks as dependencies between vehicles and task areas. In the mathematical model, the vehicle set $V = \{V_1, V_2, \ldots, V_n\}\, n \in \mathrm{N}$, and the task area set $P = \{P_1, P_2, \ldots P_n\}\, n \in \mathbb{N}$. Each task area has a fixed geographical location that can be uniquely determined by four coordinate points:

$$P_i = \{(x_{i1}, y_{i1}), (x_{i2}, y_{i2}), (x_{i3}, y_{i3}), (x_{i4}, y_{i4})\} \quad (3)$$

The availability or capacity of the task area is represented by a time slot, assuming that the start time of the time slot is t_0 and the time interval is Δt. Therefore, the set of time points at each time slot ends can be represented as:

$$T = \{t_0 + \Delta t, t_0 + 2\Delta t, t_0 + 3\Delta t, \ldots, t_0 + n\Delta t\} \quad (4)$$

Each task area corresponds to a different state S_n in each time slot t_n, with three categories of states: unavailable state N; Available state A and occupied state O.

Each vehicle V_i in the vehicle queue moves on the road network and has real-time changing geographic coordinates (x_i, y_i), which are limited by the topology of the road network and traffic rules. In addition, each vehicle has the following four core attributes:

1. Resource Ownership VR_i: Each traffic task is related to a particular attribute of the vehicle, such as cargo capacity and battery level, which are collectively referred to as resources in this article, represented as $V\mathrm{R}_i = \left\{r_i^1, r_i^2, r_i^3, \cdots\right\}$. For each resource r_i^j, a maximum threshold $\max r_i^j$ designates adequacy in resource ownership and a minimum threshold $\min r_i^j$ signifies insufficiency.
2. Executable Status VE_i: An indicator to determine if the current vehicle V_i is capable of performing a specific task. $VE_i = \left\{e_i^1, e_i^2, e_i^3, \cdots\right\}$. The cardinality of VE_i is equivalent to that of VR_i, namely, $|VR_i| = |VE_i|$. Each element e_i^j follows a 0-1 distribution, The formula is as follows:

$$e_i^j = \begin{cases} 1 & if\ r_i^j\ minr_i^j \\ 0 & otherwise \end{cases} \quad (5)$$

3. Journey Waiting Time JL_i: Represents the time loss l_i^j for vehicle V_i during its journey due to not reaching the task area P_j. $JL_i = \left\{jl_i^1, jl_i^2, jl_i^3, \cdots\right\}$. An indicator function $I_{P_j}(V_i)$, which uses the cross product [44] to determine whether vehicle V_i has reached P_j:

$$\begin{cases} \mathrm{c}_1 = (\mathrm{x_i} - \mathrm{x_{j1}}) * (\mathrm{y_{j2}} - \mathrm{y_{j1}}) - (\mathrm{y_i} - \mathrm{y_{j1}}) * (\mathrm{x_{j2}} - \mathrm{x_{j1}}) \\ \mathrm{c}_2 = (\mathrm{x_i} - \mathrm{x_{j2}}) * (\mathrm{y_{j3}} - \mathrm{y_{j2}}) - (\mathrm{y_i} - \mathrm{y_{j2}}) * (\mathrm{x_{j3}} - \mathrm{x_{j2}}) \\ \mathrm{c}_3 = (\mathrm{x_i} - \mathrm{x_{j3}}) * (\mathrm{y_{j4}} - \mathrm{y_{j3}}) - (\mathrm{y_i} - \mathrm{y_{j3}}) * (\mathrm{x_{j4}} - \mathrm{x_{j3}}) \\ \mathrm{c}_4 = (\mathrm{x_i} - \mathrm{x_{j4}}) * (\mathrm{y_{j1}} - \mathrm{y_{j4}}) - (\mathrm{y_i} - \mathrm{y_{j4}}) * (\mathrm{x_{j1}} - \mathrm{x_{j4}}) \end{cases} \quad (6)$$

$$I_{P_j}(V_i) = \begin{cases} 1 & if c_1 * c_2 * c_3 * c_4 > 0 \\ 0 & otherwise \end{cases} \quad (7)$$

When $I_{P_j}(V_i) = 0$ and vehicle V_i is in an executable state $e_i^j = 1$ at the current time t_n, the corresponding sub-element jl_i^j in the travel waiting time JL_i will accumulate as the time slot progresses, until the vehicle reaches the task area P_j, that is:

$$I_{jl_i^j} = \begin{cases} 1 & \mathrm{if} e_i^j = 1 \mathrm{and} I_{P_j}(\mathrm{V_i}) = 0 \\ 0 & otherwise \end{cases} \quad (8)$$

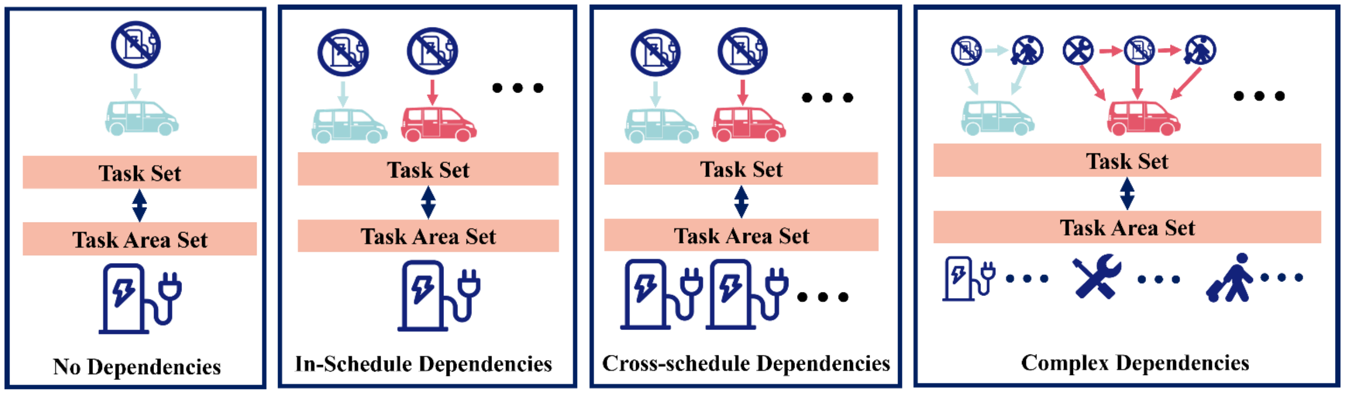


FIGURE 4. Different degrees of interdependence of Vehicle–Task utilities.

$$jl_i^j(t_n) = jl_i^j(t_{n-1}) + I_{jl_i^j}(t_n) \times \Delta t \tag{9}$$

4. Resource Waiting Time RL_i: Represents the time rl_i^j consumed by vehicle V_i while it has already arrived the task area. $RL_i = \left\{rl_i^1, rl_i^2, rl_i^3, \ldots\right\}$. Specifically, when the vehicle has already arrived at the geographical area P_j of the traffic task, with the arrival time being t_n, and its corresponding task area is in a non-available state, i.e., $S_n = O$ or $S_n = N$, this state would be represented by indicator $I_{rl_i^j}$, then the value of the corresponding sub-element in the vehicle's resource waiting time will accumulate according to the time slot. The corresponding formula is as follows:

$$I_{rl_i^j} = \begin{cases} 1 & \text{if} e_i^j = 1 \text{and} I_{P_j}(V_i) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$$rl_i^j(t_n) = rl_i^j(t_{n-1}) + I_{rl_i^j}(t_n) \times \Delta t \tag{11}$$

As vehicles engage with the road network and traffic task areas, their four aforementioned attributes undergo corresponding changes. In addition to this, the model also has the following constraints.

Define the allocation function $D\left(V_i, P_j\right)$. Each task area assigns a unique sequence number to every vehicle it encounters, and each sequence number can only be assigned to one vehicle:

$$D\left(V_i, P_j\right) = S_k$$
$$\forall V_n, V_m, V_n \neq V_m \Rightarrow D\left(V_n, P_j\right) \neq D\left(V_m, P_j\right) \tag{12}$$

Define the vehicle resource increment function $H\left(V_i, r_i^j, t_n\right)$, which indicates whether the resource quantity r_i^j increases for vehicle V_i within the time slot t_n. So, the occupation status of the task area often has a one-to-one mapping relationship with the duration of the increase in vehicle resource ownership:

$$\forall S_n = O,\ \forall t_n \sum_j S_n = \sum_i H\left(V_i, r_i^j, t_n\right) \tag{13}$$

At any given time slot t_n, the task area P_j can only be occupied by a specific vehicle V_i:

$$\forall t_n,\ \forall P_j \sum_i H\left(V_i, r_i^j, t_n\right) \leqslant 1 \tag{14}$$

Considering the mathematical model and constraints mentioned above, the optimization objective of this paper is to minimize the sum of the journey waiting time jl_i^j and resource waiting time rl_i^j of all vehicles:

$$Min \sum_i \sum_j \left(jl_i^j + rl_i^j\right) \tag{15}$$

It is important to note that this optimization objective is based on the binary assumption model of vehicles and task areas. Different task categories are all described in the same form and,therefore, have the same form of optimization objective. However, this optimization objective cannot express the interrelationships between different task categories. Taking an example of a freight vehicle V_i with insufficient battery, the time cost of the freight vehicle traveling to charging station P_1 is denoted as jl_i^1, and the waiting time within the charging station is denoted as rl_i^1. Both of these times will accumulate into the time cost of the vehicle traveling to the freight location P_2 denoted as jl_i^2. In the next section, this study will analyze in detail how to design the corresponding reinforcement learning framework based on the dependencies between tasks to solve the optimization problem.

IV. PROPOSED METHOD

This section will construct a task allocation system based on reinforcement learning. Firstly, it will analyze the relationship between the classification model proposed and reinforcement learning. Secondly, it will elaborate on the construction method of interactive simulation environments in reinforcement learning. Finally, it will construct frameworks for both single agents and multi-agent combinations. The code link of this study is:

https://github.com/naiheyudeshui/multi_agent_vehicle_scheduling.git

TABLE 2. The Relationship between Classification Models and RL.

Source	Impact on RL
Regions in binary relations	Agent entities
Vehicles in binary relations	Action space
Costs in binary relations	Reward function
Sequential dependencies within the same schedule	Event-driven asynchronous updates
Inter-task influences of Cross-Schedule Dependencies	Multi-agent architecture
Task relationships in Complex Dependencies	Different types of agents
Sources of influence in Complex Dependencies	Design of state space

A. THE CONSTRUCTION PRINCIPLE OF RL BASED ON TASK DEPENDENCY

This section will analyze the relationship between task dependency-based classification methods and Markov decision processes in reinforcement learning. The corresponding relationship is shown in the Table 2.

In reinforcement learning, an individual intelligent agent needs to gather information from the environment and make corresponding decisions. Earlier in this text, different vehicle tasks were assumed to have a binary relationship between task areas and vehicles. Different task areas typically have different environmental information, so it is reasonable to consider different task areas as distinct intelligent agents.

The action space in reinforcement learning determines how the agent interacts with the environment. Building on the analysis of binary relationships in the previous section, it is reasonable to consider a sequence of vehicles as the action space for the agent. As a result, each agent's decision-making process involves a deterministic matching between vehicles and task areas.

The reward function in reinforcement learning needs to reflect the impact of current decision steps. Based on the mathematical model in the previous context, it can be understood that each cost in the binary relationship actually reflects the journey waiting time jl_i^j and the waiting time in the region rl_i^j. Therefore, the reward function needs to be related to these parameters.

In In-Schedule Dependencies, the impact of each cost in the current binary relationship originates from the impacts brought by previous decision steps rather than from changes in the environmental dimension over time. Therefore, the decision steps of reinforcement learning in this study adopt event-driven asynchronous updates [48].

In Cross-Schedule Dependencies, areas with similar tasks will influence each other during decision-making. Therefore, it is necessary to construct multi-agent reinforcement learning to describe this influence by summing the values of multiple agents within the same decision step.

In Complex Dependencies, tasks of different types will interact with each other to become constraint conditions. Therefore, it is necessary to construct multiple types of intelligent agents based on the types of tasks to differentiate the sources of reward for different types of tasks.

In Complex Dependencies, tasks of different types are influenced by different sources. Therefore, tasks of different types require different state spaces to be designed. The state space needs to include information about the constraints and the amount of influence in the task to facilitate the intelligent agent's value network to regress the correct model.

B. INTERACTIVE ENVIRONMENT FOR RL

The simulation environment is an essential part of reinforcement learning because it requires interactive trial-and-error to optimize its own value and policy networks. This study built a complete set of simulation environments suitable for reinforcement learning based on SUMO. During training, the proposed multi-agent reinforcement learning architecture can interact in real time and obtain corresponding data. By changing the map and vehicle tasks, this simulation system can simulate vehicle dispatching in different scenarios. Through adjusting the reinforcement learning algorithm, it can train a multi-agent system adapted to the specific scenario. The following section will further explain the construction method of reinforcement learning in this study based on a specific problem.

In order to reduce the impact of road complexity on the speed of simulation software and improve the efficiency of data acquisition by intelligent agents, this study selected large industrial park roads as the basic model. Figure 5 shows a partial road network of Suzhou Industrial Park. In this study, an area covering approximately sixteen square kilometers within the park was selected as the simulation area for task allocation.

To ensure that the simulated problem types accurately reflect the complex dependencies of traffic problems mentioned earlier, it requires at least two task types, with each corresponding to multiple task areas. Additionally, mutual dependencies between task types are also necessary. Therefore, this article selects cargo transportation and charging tasks. Through investigation, it was found that there are five charging stations within the simulation area. Additionally, six locations were chosen as warehouses.

For transportation tasks, each warehouse has specific and quantity-limited parking locations. For each specific transportation task, two warehouse locations will be randomly selected, one as the loading location and the other as the unloading location. In a transportation task, vehicles need to drive from one warehouse to another in order and follow parking location restrictions. The method of setting up warehouse stops in SUMO is the same as that of bus stops, which are viewed as fixed areas on the road. The specific task information is shown in Table 3.

For charging tasks, the simulation environment includes the maximum and current battery levels of each vehicle. Each charging station contains a limited number of charging poles, which are uniformly set at 200kw. Each charging task

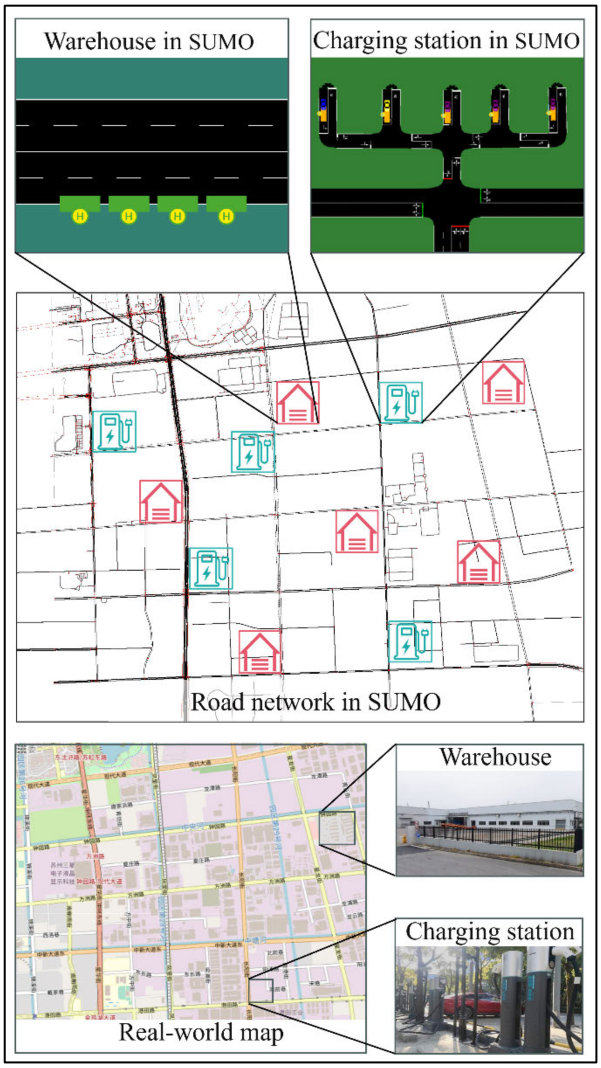


FIGURE 5. Construction of simulation environment in SUMO.

TABLE 3. Task attributes in a simulation environment.

	Transportation		Charging
Task area	Pickup warehouse	Unloading warehouse	Charging station
Task dependencies	Charging	Charging, Pickup	-
Vehicles per task	1	1	1
Parking spaces	4	4	5
Task execution time	300s	300s	600-900s

requires the vehicle to go to an idle charging station and stay until fully charged to complete. In SUMO, the implementation of charging stations and charging poles in this study follows the approach of Kurczveil et al. [49]. The specific task information is shown in Table 3.

Each vehicle in the simulation environment has random initialization parameters. The specific initial parameters of the vehicles are described in Table 4. Additionally, the

TABLE 4. Initial vehicle parameters in the simulation.

Parameters	Unit	Value
V_i Initial position coordinates	(x, y)	random
Initial battery capacity	ratio	[0.1-0.5]
Maximum battery capacity	kWh	[30-50]
Upper threshold $maxr_i^j$	ratio	1
Lower threshold $minr_i^j$	ratio	0.3
Journey waiting time JL_i	seconds	{0, 0}
Resource waiting time RL_i	seconds	{0, 0}
Air drag coefficient	dimensionless	0.35
Maximum speed	m/s	25
Minimum gap	meters	1.5
Acceleration	m/s^2	2.0
Deceleration	m/s^2	2.5
Internal moment of inertia	$kg \cdot m^2$	0.01
Recuperation efficiency	percentage	[80%-96%]
Roll drag coefficient	dimensionless	0.02
Radial drag coefficient	dimensionless	0.1
Vehicle mass	kg	[1500-2500]

Krauss [50] car-following model, LC2013 [51] lane-changing model, and Dijkstra [52] path-searching model are employed.

C. CONSTRUCTION OF AGENTS FOR CHARGING STATIONS

The action space of each charging station intelligent agent needs to include its matching relationship with all vehicles. As analyzed earlier in the paper, the task assignment problem in traffic essentially involves matching task vehicles with task areas. Furthermore, since this paper regards charging stations as intelligent agent objects, their action space needs to determine which vehicle to assign a charging task to under the current situation or decide not to assign any charging tasks. Therefore, their action space comprises 51 discrete actions. When it chooses to schedule a charging task for a certain vehicle, the vehicle will go to the current charging station to perform the charging task. When all charging poles in the charging station are fully occupied, the vehicle will queue up and wait.

The objective of the charging station agent is to ensure all task vehicles maintain sufficient battery levels while minimizing the time lost due to charging. To achieve this, the charging station agent needs to continuously monitor the current battery status of each vehicle in real time and schedule charging tasks for vehicles with low battery levels. Additionally, to minimize the time lost due to charging, the charging station agent needs to reduce the frequency of vehicle charging, aiming to schedule charging tasks when the vehicle battery levels are low. Furthermore, the charging station agent needs to minimize the waiting time for vehicles to reach the charging station (denoted as JL_i), by directing vehicles to the nearest available charging station whenever possible. Moreover, the charging station agent needs to minimize the waiting time for vehicles to access charging resources (denoted as RL_i), by scheduling charging tasks to avoid congestion at the charging station.

In order to achieve the goals of the charging station intelligent agent mentioned above, the following reward function has been set:

$$R_1 = \sum_{i}^{M} \left(50 \times ln\left(r_i^1 + 0.3\right) + 20\right) \tag{16}$$

where, M represents the set of vehicles that have not been assigned charging tasks, r_i^1 denotes the current battery level ratio of vehicle V_i. This reward function reflects the attention paid to the battery status of all vehicles. When a vehicle has a low battery level and has not been assigned a charging task, the intelligent agent will receive a relatively large negative reward.

$$R_2 = 500 \times N - \sum_{i=1}^{N} \left(j_i^1 + r_i^1 + j_i^2\right) \tag{17}$$

where N represents the set of vehicles that have been assigned charging tasks, j_i^1 denotes the time vehicle V_i consumes when traveling to the charging station, r_i^1 represents the time vehicle V_i spends waiting in queue at the charging station, and j_i^2 represents the time vehicle V_i consumes on the way to complete transportation tasks. Here, it indicates the extent to which traveling to the charging station affects the original transportation tasks.

Apart from the aforementioned rewards, it is essential to ensure that multiple agents do not conflict over the allocation of a particular vehicle. Such conflicts encompass two scenarios: firstly, assigning charging tasks to vehicles that already have charging arrangements, and secondly, multiple agents simultaneously assigning charging tasks to the same vehicle in a single decision step. Therefore, this paper implements a reward adjustment mechanism based on auction theory. At each decision step, if conflicts arise involving the aforementioned two scenarios, the central controller first estimates the total waiting time that vehicles may experience at different charging stations to complete their charging tasks. Based on this estimation, the central controller then adjusts the rewards for the charging station agents accordingly.

$$T(i,j) = T_{jl}(i,j) + T_{rl}(i,j) \tag{18}$$
$$R_3 = 50 \times (Min(T(i,j)) - T(i,j)) \tag{19}$$

where $T(i,j)$ represents the estimated total waiting time for vehicle V_i to complete its charging task at the charging station P_j, encompassing the anticipated travel waiting time T_{jl} and the anticipated queue waiting time T_{rl} at the charging station. Among all participating charging stations in the task competition, the shortest estimated time is denoted as $\min(T(i,j))$. In each iteration step, the rewards obtained by the charging station intelligent agent are the sum of the above rewards, namely:

$$R_{\text{total}} = R_1 + R_2 + R_3 \tag{20}$$

The state space of the charging station agent includes real-time information on each vehicle and the current situation of each charging station. For each vehicle, it includes the current battery level, real-time distances to all charging stations and warehouses, the executable status of the current charging task, and the executable status of the current transportation task. The charging station situation includes the future idle time of the five charging poles at each charging station. Therefore, the total dimensionality of the state space is 725.

D. CONSTRUCTION OF AGENTS FOR WAREHOUSES

The action space dimensionality of the warehouse agent is the same as that of the charging station agent, which includes 50 discrete dimensions for matching specific vehicles and one dimension for indicating no action at the current decision step. Whenever a warehouse agent selects a task vehicle, the task vehicle needs to proceed to the designated warehouse for cargo loading, and the designated unloading warehouse is automatically set as the next destination.

The objective of the warehouse agent is to schedule vehicles to complete all current delivery tasks while maximizing delivery efficiency. To achieve this goal, the warehouse agent needs to monitor vehicles that have not been assigned tasks in real time to utilize each vehicle's transportation capacity as much as possible. Furthermore, the warehouse agent needs to prioritize selecting vehicles with sufficient battery levels to ensure that vehicles do not require charging during a single transportation process. Additionally, the warehouse agent needs to assign cargo tasks to vehicles closer to itself to minimize time loss during cargo transportation.

In order to achieve the above goals, the warehouse agent has the following reward function:

$$R_1 = -100 \times \sum_{i}^{50} e_i^2 \tag{21}$$

Here, e_i^2 denotes whether a vehicle is in an empty state. The more vehicles that are empty, the more negative rewards are obtained by the warehouse agent.

$$R_2 = 500 \times N - \sum_{i=1}^{N} \left(j_i^2 + r_i^2\right) \tag{22}$$

Here, N represents the number of vehicles already assigned transportation tasks, j_i^2 is the time consumed by vehicle V_i when traveling to the warehouse, and r_i^2 is the time vehicle V_i spends waiting in the queue at the warehouse.

Apart from the aforementioned rewards, during the interaction process among multiple warehouse agents, conflicts may arise due to the competition for specific vehicles. These conflicts encompass two scenarios: firstly, assigning new transportation tasks to vehicles that already have ongoing transportation tasks; secondly, multiple agents simultaneously assigning transportation tasks to the same vehicle in a single decision step. Therefore, similar to the setup for the charging station agents in this study, a reward adjustment mechanism is implemented.

$$T(i,j) = T_{jl}(i,j) + T_{rl}(i,j) \tag{23}$$
$$R_3 = 50 \times (Min(T(i,j)) - T(i,j)) \tag{24}$$

Algorithm 1 Single agent decision-making

```
1  Agent Charging station $P_i$ :
       Input: State space $S_t$ =[Current vehicle
       information;
       Current charging station information]
2      for $\forall S_t$ do
3          $A_t \sim \pi_{P_i}\left(\cdot \mid S_t;\theta^i\right)$
       Output: $A_t$
       //(Specify a vehicle to perform a transportation
       task, or chooses not to take any action.)
4  Agent Warehouse $P_j$:
       Input: State space $S_t$, = [The specific information
       of
       the current transportation task; Current vehicle
       information; Current warehouse information]
5      for $\forall S_t$ do
6          $A_t \sim \pi_{P_j}\left(\cdot \mid S_t;\theta^j\right)$
       Output: $A_t$
       //(Specify a vehicle to perform a transportation
       task, or chooses not to take any action.)
```

Here, $T\ (i, j)$ represents the estimated total waiting time for vehicle V_i to complete a single transportation task, encompassing the estimated travel waiting time T_{jl} and the estimated queue waiting time T_{rl} at the warehouse. Among the conflicting transportation tasks, the shortest estimated time is denoted as Min $(T\ (i, j))$.

In each iteration step, the rewards received by the warehouse agent are the sum of the above rewards, namely:

$$R_{total} = R_1 + R_2 + R_3 \tag{25}$$

The state space of each warehouse agent needs to include the specific information of the current transportation task, which comprises the distance information to the destination and the estimated energy consumption of the transporting vehicle. Secondly, it includes the total number of pending transportation tasks for all warehouse agents. Finally, it encompasses the specific information of all task vehicles, including their current battery level, current cargo status, and distance information to the six warehouses. The total state space is 407.

E. MULTI-AGENT RL GENERAL FRAMEWORK

This framework utilizes multi-agent reinforcement learning to address the traffic task allocation problem with complex dependencies. The construction of individual agent has been provided in the preceding sections (overview as shown in Algorithm 6). When multiple agents collaborate, this paper categorizes them into two groups: warehouse agent group and charging station agent group, as illustrated in Figure 6. The policy network of the i-th agent can be represented as follows:

$$\hat{f} = \pi\left(\cdot \mid s;\theta^i\right) \tag{26}$$

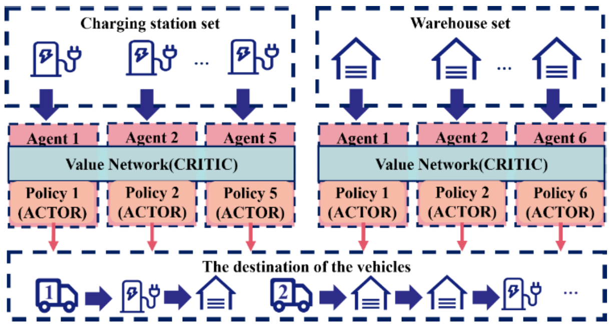


FIGURE 6. Framework diagram of multi-agent reinforcement learning.

Here, s represents the input state of the policy network, $\hat{f}$ is the output vector whose dimension corresponds to the size of the action space $|A^i|$. Furthermore, $\hat{f}$ signifies the probability of each action output.

Due to each agent having its own policy network, the actions of all agents at state S_t can be represented as $A_t = \left[A_t^1, \ldots, A_t^m\right]$. Additionally, since all agents collectively influence the environment, the future returns U_t^i of agent i also depend on the future action sequences of all resource agents $[A_t, A_{t+1}, A_{t+2}, \ldots, A_n]$. However, U_t^i is still a random variable at this point. If, based on the observed A_t, and the expectation is utilized to eliminate unknown actions and states after time $t+1$ in U_t^i, the action-value function for that agent can be obtained:

$$Q_\pi^i\ (s_t, a_t) = E_{S,A}\left[U_t^i \mid S_t = s_t, A_t = a_t\right] \tag{27}$$

where s_t represents a fixed state among all possible states, and a_t represents a fixed combination of actions among all possible combinations in A_t.

The policy gradient theorem in reinforcement learning, further extended with the introduction of the baseline algorithm, is as follows:

$$\begin{aligned}&\nabla_{\theta^i} J\left(\theta^1, \cdots, \theta^m\right)\\&= E_{S,A}\left[(Q_\pi\ (S, A) - b)\cdot \nabla_{\theta^i} ln\pi\left(A^i \mid S;\theta^i\right)\right]\end{aligned} \tag{28}$$

Where $\nabla_{\theta^i} J\left(\theta^1, \cdots, \theta^m\right)$ represents the policy gradient of the i-th agent's policy parameters θ^i; $E_{S,A}$ denotes the expectation over future states S and actions A; b is a baseline introduced to reduce variance during training, thus improving the stability and efficiency of learning, typically represented by the state value function $V_\pi\ (s)$; $\nabla_{\theta^i} \ln\pi\left(A^i \mid S;\theta^i\right)$ is the logarithmic policy gradient, representing the gradient of agent i's policy with respect to its parameters θ^i. Based on the policy gradient theorem, the policy network parameters of the i-th agent can be updated, thereby facilitating continuous improvement of all agents' policies during the training process. An overview of multi-agent interaction is shown in Algorithm 16.

V. EXPERIMENTAL RESULTS AND ANALYSIS

This study proposes a multi-agent reinforcement learning approach that enables real-time solutions to the scheduling

Algorithm 2 Multi agent interaction strategy

```
Data: Real-time vehicle data; Real-time warehouse data;
      Real-time charging station data
Result: Send scheduling information to vehicle $V_i$
1  Function Single iteration step:
2      Database $\overset{update}{\longleftarrow}$ Real-time data;
3      Rewards $\overset{update}{\longleftarrow}$ Real-time data;
4      for Charging station $P_i$ in Agent set do
5          Update the specified state space $S_t^i$ of the agent;
6          Schedule Charging station agent $P_i$ and obtain $A_t^i$;
7      for Warehouse $P_j$ in Agent set do
8          Update the specified state space $S_t^j$ of the agent;
9          Schedule Warehouse agent $P_j$ and obtain $A_t^j$;
10     $A_t = \left[A_t^1, \ldots, A_t^m\right]$;
11     if scheduling conflicts occur in the task set $A_t$ then
12         Evaluate all conflicting tasks and obtain $Min(T(i,j))$;
13         Apply additional negative rewards to conflicting agents;
14         Update $A_t$ based on evaluation results;
15     Update policy gradient $\nabla_{\theta^i} J\left(\theta^1, \ldots, \theta^m\right)$;
16     Send scheduling information to vehicle $V_i$;
```

problem of multiple vehicles and multiple tasks. In order to verify its performance, this section will conduct experimental comparisons from multiple perspectives and analyze the reasons for the results.

A. DESCRIPTION OF TEST CASES

With the emergence of autonomous driving vehicles, maintenance-free, multi-task integration, real-time capabilities, and the ability to dispatch multiple vehicles will be an inevitable trend for future systems. However, at present, this field still lacks a mature theory and a universal method [53]. Especially in the timing of switching between multiple tasks, current scheduling algorithms often lack adaptability. Therefore, in designing the comparative algorithm in this paper, a fixed rule task-switching method was adopted, whereby when the battery level of a task vehicle drops below 30%, the system will switch the vehicle's task to charging. When the battery level is above 30%, the system will switch the vehicle's mission to transportation. Within a single task, this study has designed two scheduling methods as comparative algorithms:

1) FCFS: When the vehicle's battery level is above 30%, the vehicle will be assigned a transportation task from the nearest warehouse. When the vehicle's battery level is below 30%, the vehicle will go to the nearest charging station to complete the charging task [54].
2) Digraph: In the scheduling strategy, a directed graph can incorporate all time-related variables as edge weights [55]. Building upon the FCFS strategy, this approach takes into account the occupancy of charging stations and warehouses as road weights, utilizing the Dijkstra algorithm to determine the optimal path to guide vehicles in task pickups.

As a comparison, the training parameters of the multi-agent reinforcement learning multi-task allocation method in this article are shown in Table 5.

TABLE 5. Hyperparameters.

Types	Hyperparameter	Value
Charging station	Value network layers	15
	Policy network layers	15
	Action matrix size	51
	State matrix size	725
	Replay buffer size	10,000
	Minibatch size	256
	Learning rate	0.0001
	Discount factor γ	0.8
Warehouse	Value network layers	13
	Policy network layers	13
	Action matrix size	51
	State matrix size	407
	Replay buffer size	10,000
	Minibatch size	256
	Learning rate	0.0001
	Discount factor γ	0.8

This paper will analyze the simulation from two perspectives. The first perspective will consider the efficiency of completing tasks from a holistic perspective. The second perspective will provide a more detailed analysis of the differences in task completion efficiency between rule-based algorithms and MARL algorithms.

During the simulation, the frequency of transport tasks affects the overall performance of multi-task allocation. At higher task frequencies, queues may form at warehouses and charging stations, leading to an increase in the average task completion time. Therefore, higher task frequencies also test the scheduling performance of algorithms. A parameter is set in this paper to describe this frequency, namely the number of tasks per square kilometer per hour. All subsequent experiments in this article will involve this parameter.

The entire verification was implemented on a computer equipped with an i7-14700 processor, a 4090 graphics card, and the Ubuntu 22.04 operating system. In addition, this simulation used SUMO as the underlying environment, Python as the programming language to interact with it, and implemented a neural network architecture based on PyTorch.

B. TASK COMPLETION EFFICIENCY

This study first attempts to fix the frequency of transportation tasks in the simulation environment. Figure 7 records

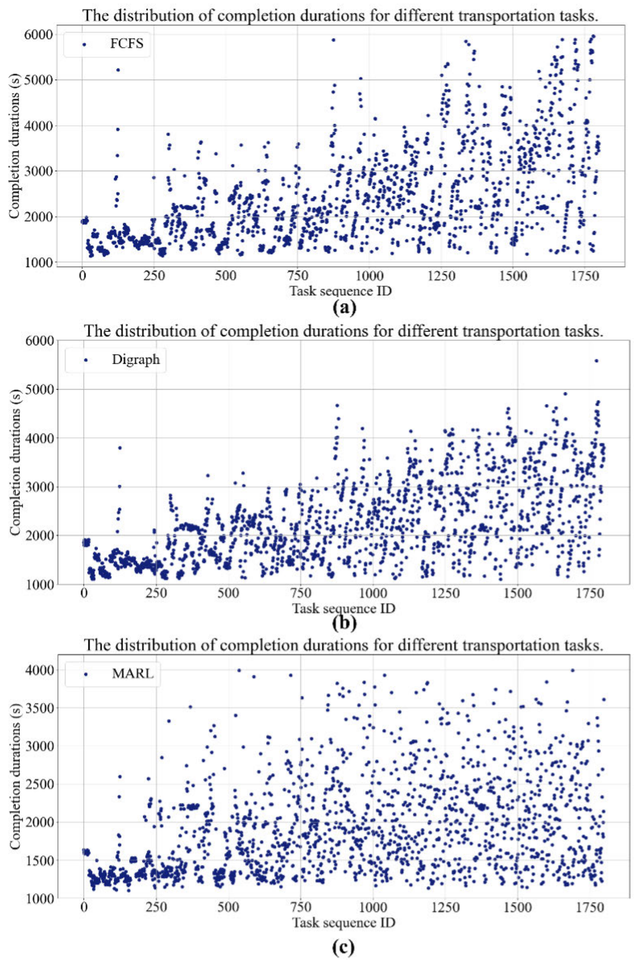


FIGURE 7. The distribution of completion durations for different transportation tasks. (a) FCFS (b) Digraph (c) MARL.

the completion times of different order tasks under different scheduling algorithms, corresponding to a frequency of 24 transportation tasks per square kilometer per hour. Each scheduling algorithm was tested continuously with 1800 sets of transportation tasks and the corresponding scatter plots were provided.

As shown in Figure 7, the completion times of the three algorithms for the initial hundreds of tasks are relatively short, and the time required for the three scheduling algorithms to complete the tasks is also similar. This is because, in the current situation, the vehicles have sufficient battery power and do not need to charge. Additionally, most vehicles are initially empty, allowing them to respond to transportation tasks in real-time.

With the ongoing transportation tasks, the three algorithms showed differences in completion time. Clearly, in the FCFS algorithm, the deviation in task completion time gradually increased, with a significant fluctuation range in statistical data. Some transportation tasks even required over 6000 seconds to complete. Compared to FCFS, the directed graph reduces peak values significantly, resulting in a substantial decrease in the number of tasks completing after 4000 seconds. However, there is still a wide range of completion times for tasks, with an increasing deviation. In contrast, the MARL algorithm proposed in this paper demonstrates good robustness, with completion times for most tasks under 2500 seconds. Moreover, as transport tasks continue to emerge, completion times remain stable, with few requiring over 4000 seconds.

Due to the fact that a directed graph can comprehensively consider all selection spaces, it can always make better choices at the current time step compared to FCFS. However, neither FCFS nor the directed graph takes into account the dependencies between tasks before and after making decisions, resulting in an increasing deviation in task completion times for both algorithms. The MARL proposed in this paper, on the other hand, can integrate the influence between tasks before and after, equalize the utilization of task intervals, and keep the completion time of tasks within a stable range.

Additionally, this study divides the completion time of order tasks into multiple intervals. The results of three task allocation methods were statistically analyzed, as shown in Figure 8. This figure can more intuitively demonstrate the stability of the Multi-Agent Reinforcement Learning algorithm proposed in this paper in task allocation problems, which minimizes the number of high time-consuming tasks as much as possible. Most tasks have completion times concentrated between 1000 and 2000 seconds, and the distribution of task times is relatively uniform. In contrast, in the FCFS algorithm, the distribution of task completion times is the most uneven and covers a wide span on the statistical chart. The algorithm for directed acyclic graphs falls between the two, reducing the workload of high-consumption tasks but still having many tasks with completion times exceeding 3000 seconds. This graph also demonstrates that, compared to FCFS, directed acyclic graphs can better optimize tasks with high consumption time, but can only make the best decision at the current time step, unable to coordinate multiple tasks from a temporal perspective. Therefore, the overall optimization effect is not as good as Multi-Agent Reinforcement Learning (MARL).

In order to further explore the scheduling capabilities of algorithms at different task frequencies, this study established a simulation environment with different task frequencies. Specifically, these three algorithms need to schedule vehicles to complete 2000 order tasks in a simulation environment with 10 transport tasks per square kilometer per hour. Subsequently, the task frequency in the simulation environment gradually increases, and the three scheduling algorithms still need to complete 2000 tasks each at different task frequencies. Finally, the transport task frequency in the simulation environment will reach 42 order tasks per square kilometer per hour. Furthermore, this study aggregated and calculated the average completion times of these three algorithms across 2000 tasks, as shown in Figure 9.

From the data graph, it can be further observed that the proposed MARL algorithm exhibits better resilience under high task densities. As task density increases, the increase in average completion time is relatively gradual. Moreover,

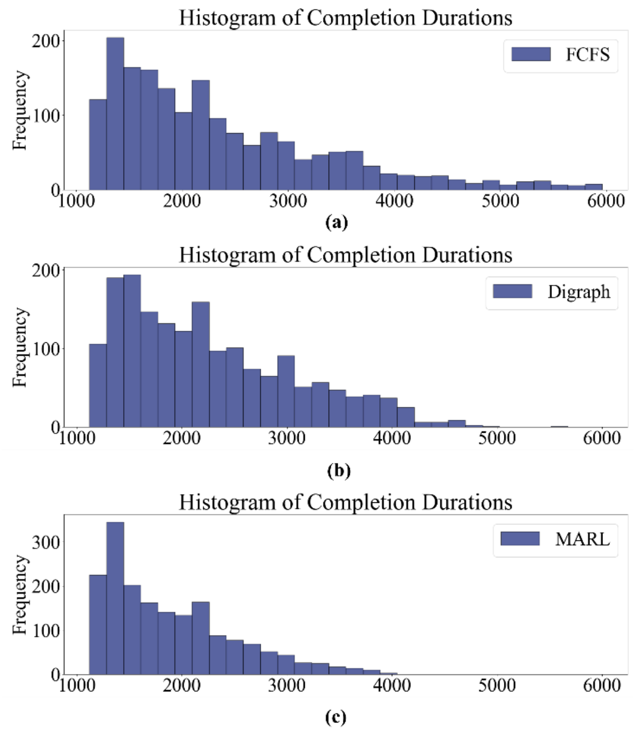


FIGURE 8. **Histogram of completion durations. (a) FCFS (b) Digraph (c) MARL.**

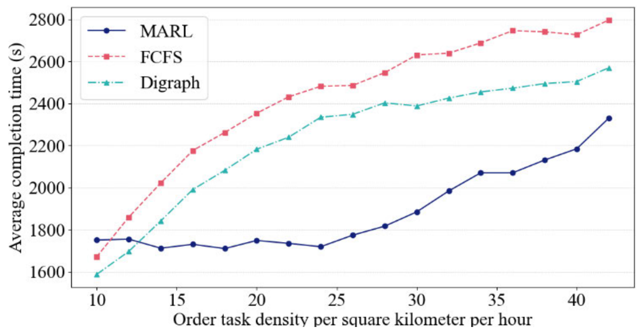


FIGURE 9. **The average completion time under different transportation task frequencies.**

when the task density is below 25, there is no significant fluctuation in task completion time. In contrast, for FCFS and Digraph, as task density increases, the average completion times of their ordered tasks show a more pronounced increasing trend, with the Digraph algorithm consistently outperforming the FCFS algorithm. With the increase in order density, the performance gap between FCFS and Digraph widens. However, surprisingly, when task density is low, the performance of the MARL algorithm proposed in this paper is inferior to the other two rule-based algorithms. The following sections of this paper will further explore the reasons behind this phenomenon.

C. REASONS FOR DIFFERENCES

This section will analyze the reasons for the differences in task allocation of three algorithms from three perspectives. The first perspective is the relationship between task scheduling and task area utilization rate, the second perspective is the relationship between the timing of task matching and the state of the vehicle, and the third perspective is the sequential relationship between task scheduling.

For the utilization rate of the task area, this study further set two different transportation task densities, which are 10 transportation tasks per square kilometer per hour and 26 transportation tasks per square kilometer per hour. Under these two task densities, the total power of all charging stations in the simulation area was recorded, which can represent the utilization rate of charging stations. The results are shown in Figure 10.

Under the lower task density, we observed that the charging demand in the area did not reach the capacity limit, meaning that some charging stations were idle. In this scenario, focusing on optimizing path distances, the FCFS algorithm can effectively reduce waiting times for vehicles, with minimal differences between the Digraph method and the FCFS method. In contrast, the Multi-Agent Reinforcement Learning (MARL) framework seems to prefer evenly distributing tasks at each time interval, maintaining the overall utilization of charging stations at lower levels. Thus, some vehicle charging tasks may be scheduled at charging stations further away. This allocation strategy results in a significant number of vehicles still charging in the time period of 3000-3500 seconds, as shown in Figure 10(a). This also explains why in Figure 9, when the order task frequency is low, the average waiting time of the MARL algorithm is higher than that of the FCFS and Digraph methods. However, the benefits of doing so are also very clear, as this task allocation method will provide redundant charging resources for task areas with high task density.

At the high task density shown in Figure 10(b), the vehicles' charging demand has already reached the capacity limit of the charging station. In this scenario, a more uniform task allocation method actually helps improve the utilization of task areas, thereby reducing the average waiting time of vehicles and further enhancing the overall efficiency of task completion. This suggests that the MARL architecture proposed in this paper is more effective in dealing with high task density allocation issues, leading to an improvement in overall system efficiency. In contrast, the Digraph algorithm, at around 2000 seconds, improves the charging station's utilization through scheduling, thereby shortening the average waiting time of vehicles at the charging station. This is also why the Digraph algorithm is always superior to the FCFS algorithm.

In addition to studying the utilization rate of the task region, this article also delves into the inherent connection between the timing of task matching and the vehicle's status. The scheduling agent must accurately utilize vehicle status information to determine the optimal timing for executing charging tasks reasonably. If the charging task is scheduled too late, the vehicle's battery may have depleted to a very low level, making it impossible for the remaining power to support the vehicle's arrival at the charging station. On the

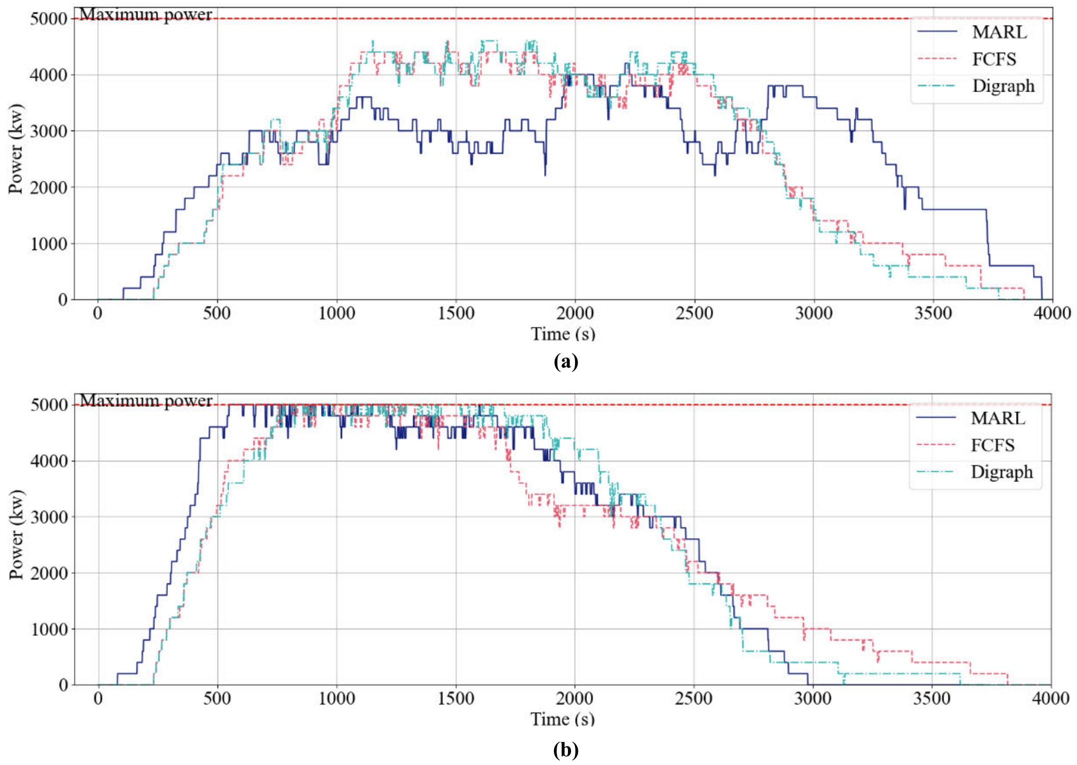


FIGURE 10. Charging station utilization based on grid power. (a) 10 transportation tasks per hour per square kilometer. (b) 24 transportation tasks per hour per square kilometer.

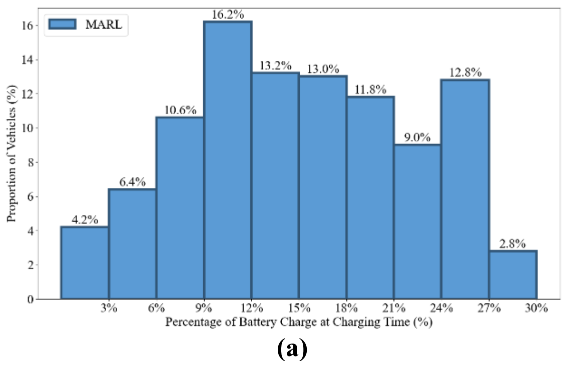
contrary, if a charging task is arranged when the vehicle's battery still has sufficient power, it will result in unnecessary loss of transportation time. Therefore, accurately balancing the match between the vehicle condition and task timing is crucial for optimizing scheduling efficiency and enhancing vehicle operation effectiveness.

In order to evaluate whether the MARL system makes full use of vehicle information in task assignment, 10,000 charging tasks were recorded in this study, and the remaining power of the vehicle when it actually started charging was calculated. This paper divides the remaining amount of electricity into different intervals and summarizes the statistical results in Figure 11. Although FCFS and Digraph have different decision-making methods for handling a single task, they both follow a fixed rule when switching tasks. Therefore, in comparison, this study refers to these two types collectively as Rule-based.

Experimental results show that compared with the rule-based method, the power of the vehicle dispatched by the MARL method is generally lower during charging. This phenomenon has led to a reduction in the frequency of charging of vehicles dispatched by MARL, which in turn has increased delivery times and improved delivery efficiency. Specifically, nearly 80% of vehicles are between 10% and 30% charged at the time of charging, demonstrating that the MARL system can efficiently use the vehicle's own status information for scheduling. However, it is worth noting that about 10% of vehicles will only be charged when the battery level drops below 6%, which exposes the need for the MARL system to add a charging protection mechanism to ensure that the vehicle does not fall into a low battery situation during task assignment, so as to ensure the stability and safety of the system. On the other hand, the charging scheduling of the rule-based algorithm will intervene and arrange the charging task when the power of the vehicle is less than 30% after completing the order, so the power of most vehicles during charging is maintained within a fixed range.

Furthermore, this study also investigated the sequential relationships between task scheduling. Due to the division of MARL agents into the charging station agent group and warehouse agent group in this research, the action spaces of these two groups of agents are independent of each other. Therefore, when warehouse agents and charging station agents assign tasks to vehicles, different state combinations may occur.

This study documents the sequence of vehicles with both transportation and charging tasks going to their destinations. The experimental results are shown in Figure 12. These results further illustrate that the MARL architecture can flexibly utilize the order of tasks during the task allocation process, rather than strictly following a set of fixed rules. This flexibility can improve the overall task completion efficiency. For instance, during transportation, if a vehicle with insufficient battery power happens to be near a charging station,



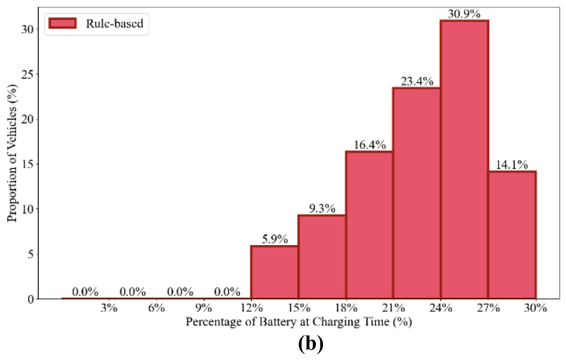


FIGURE 11. The proportion of remaining battery during task vehicle charging (a) MARL (b) Rule-based.

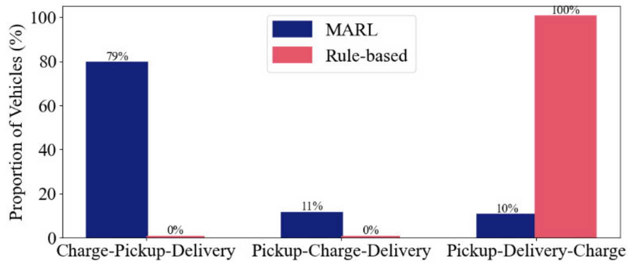


FIGURE 12. The proportion of task sequence for vehicles.

adopting the strategy of ''Pickup-Charge-Delivery'' may save more time for the vehicle.

D. FUTURE OUTLOOK

The previous experiments have preliminarily demonstrated the feasibility of the framework proposed in this study. The goal of this research is to provide a universal solution for real-time scheduling problems involving multiple vehicles and tasks. However, the current experiments only show its feasibility in specific scenarios.

In this section, we will briefly discuss the generalizability, scalability, and future challenges of applying this research in other multi-vehicle multitasking scenarios.

From the perspective of generalizability, it is necessary to analyze whether this framework can be applied to a wider range of task allocation problems. The core idea of this study is to use the state value function in reinforcement learning to describe the degree of interaction between tasks. In this study, there are interactions between different types of tasks, and the interactions within the same decision step can be measured by instantaneous rewards during multi-agent games. The mutual influence of different decision steps can be measured using the Bellman equation. Similarly, the state-value function in reinforcement learning is also derived from instantaneous rewards and the recursive Bellman equation. Therefore, if the impact sources of vehicle tasks can be detailed systematically, the framework proposed in this study can be applied. Theoretically, the framework has good generalizability.

From the perspective of scalability, it is essential to further analyze how the application of this research to a larger scale would impact. This larger scale can be analyzed from two perspectives: more vehicles and larger geographical areas.

From the perspective of vehicles, there are two ways to scale up the size of the vehicles in this study. The first approach is to set a large initial capacity for the state space during training, so that the number of vehicles it can handle matches the scale of the problem. The second approach is to design a filtering mechanism for the state space, only providing the agent with vehicles that are more relevant to the current scheduling (for example, selecting the closest 50 vehicles to that area). This is because the state space of the intelligent agent is not directly tied to specific vehicles in the action space, but rather describes parameters of different vehicles. By matching the vehicle selection in action space with the vehicle parameters in state space, the agent can make decisions in a larger scale of vehicle fleet.

From a regional perspective, a larger region implies a greater number of intelligent agents, but a higher number of intelligent agents is actually disadvantageous for the convergence of multi-agent reinforcement learning. Therefore, this study suggests that when the regional scope expands, a hierarchical decomposition of the region can be conducted, mapping regions of different sizes to intelligent agents at different levels, thereby combining multi-agent reinforcement learning with hierarchical reinforcement learning [56]. This may be a promising approach, but further experiments are needed in the future for a more detailed exploration.

From a future perspective, there are still areas for improvement in the current framework. For example, it needs to be combined with fixed rules and constraints to enhance the interpretability of reinforcement learning decisions. Additionally, the core idea of this study is to use the state-value function in reinforcement learning to describe the degree of interaction between tasks. Therefore, how to construct the structure of the value function, how to design the training method to make the description of future states more accurate is still a question worth pondering.

VI. CONCLUSION

This study proposes a binary vehicle and task-area assumption and a vehicle task classification model based on this assumption. The classification model provides a theoretical basis for the multi-agent reinforcement learning constructed in this study. After constructing a simulation environment for the transportation by electric trucks in SUMO, the proposed method is specifically applied. Experimental results demonstrate that the method effectively improves the overall efficiency of tasks and exhibits good robustness under high task density conditions.

REFERENCES

[1] M. M. Rana and K. Hossain, "Connected and autonomous vehicles and infrastructures: A literature review," *Int. J. Pavement Res. Technol.*, vol. 16, no. 2, pp. 264–284, Mar. 2023, doi: 10.1007/s42947-021-00130-1.

[2] J. Zhao, H. Hu, Y. Han, and Y. Cai, "A review of unmanned vehicle distribution optimization models and algorithms," *J. Traffic Transp. Eng., English Ed.*, vol. 10, no. 4, pp. 548–559, Aug. 2023, doi: 10.1016/j.jtte.2023.07.002.

[3] J. Ren, C. Ye, and F. Yang, "Research on path optimization modeling and algorithm of workshop handling robot with time window," *Oper. Res. Manage. Sci.*, vol. 29, no. 5, pp. 52–60, 2020, doi: 10.12005/orms.2020.0118.

[4] S. M. Meshkani and B. Farooq, "Distributed ride-matching for shared ridehailing service with intelligent city infrastructure," in *Proc. IEEE Int. Smart Cities Conf. (ISC2)*, Pafos, Cyprus, Sep. 2022, pp. 1–6, doi: 10.1109/ISC255366.2022.9921875.

[5] Z. Su, Y. Hui, and T. H. Luan, "Distributed task allocation to enable collaborative autonomous driving with network softwarization," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2175–2189, Oct. 2018, doi: 10.1109/JSAC.2018.2869948.

[6] X. Chen, S. He, Y. Zhang, L. Tong, P. Shang, and X. Zhou, "Yard crane and AGV scheduling in automated container terminal: A multi-robot task allocation framework," *Transp. Res. C, Emerg. Technol.*, vol. 114, pp. 241–271, May 2020, doi: 10.1016/j.trc.2020.02.012.

[7] D. Mo, X. Chen, and J. Zhang, "Modeling and managing mixed on-demand ride services of human-driven vehicles and autonomous vehicles," *Transp. Res. B, Methodol.*, vol. 157, pp. 80–119, Mar. 2022, doi: 10.1016/j.trb.2022.01.003.

[8] C. Dinelli, J. Racette, M. Escarcega, S. Lotero, J. Gordon, J. Montoya, C. Dunaway, V. Androulakis, H. Khaniani, S. Shao, P. Roghanchi, and M. Hassanalian, "Configurations and applications of multi-agent hybrid drone/unmanned ground vehicle for underground environments: A review," *Drones*, vol. 7, no. 2, p. 136, Feb. 2023, doi: 10.3390/drones7020136.

[9] M. Ostermeier, A. Heimfarth, and A. Hübner, "Cost-optimal truck-and-robot routing for last-mile delivery," *Networks*, vol. 79, no. 3, pp. 364–389, Apr. 2022, doi: 10.1002/net.22030.

[10] M. D. Dean, K. M. Gurumurthy, F. de Souza, J. Auld, and K. M. Kockelman, "Synergies between repositioning and charging strategies for shared autonomous electric vehicle fleets," *Transp. Res. D, Transp. Environ.*, vol. 108, Jul. 2022, Art. no. 103314, doi: 10.1016/j.trd.2022.103314.

[11] K.-F. Chu, A. Y. S. Lam, and V. O. K. Li, "Joint rebalancing and vehicle-to-grid coordination for autonomous vehicle public transportation system," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 7156–7169, Jul. 2022, doi: 10.1109/TITS.2021.3067044.

[12] X. Zhou, G. Wei, Y. Zhang, Q. Wang, and H. Guo, "Optimizing multi-vehicle demand-responsive bus dispatching: A real-time reservation-based approach," *Sustainability*, vol. 15, no. 7, p. 5909, Mar. 2023, doi: 10.3390/su15075909.

[13] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Comput. Surv.*, vol. 52, no. 2, pp. 1–31, Mar. 2020, doi: 10.1145/3303848.

[14] H. Chakraa, F. Guérin, E. Leclercq, and D. Lefebvre, "Optimization techniques for multi-robot task allocation problems: Review on the state-of-the-art," *Robot. Auto. Syst.*, vol. 168, Oct. 2023, Art. no. 104492, doi: 10.1016/j.robot.2023.104492.

[15] J. P. Queralta, J. Taipalmaa, B. C. Pullinen, V. K. Sarker, T. N. Gia, H. Tenhunen, M. Gabbouj, J. Raitoharju, and T. Westerlund, "Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision," *IEEE Access*, vol. 8, pp. 191617–191643, 2020, doi: 10.1109/ACCESS.2020.3030190.

[16] L. A. Silva, H. S. S. Blas, D. P. García, A. S. Mendes, and G. V. González, "An architectural multi-agent system for a pavement monitoring system with pothole recognition in UAV images," *Sensors*, vol. 20, no. 21, p. 6205, Oct. 2020, doi: 10.3390/s20216205.

[17] D. Muravev, H. Hu, A. Rakhmangulov, and P. Mishkurov, "Multi-agent optimization of the intermodal terminal main parameters by using anylogic simulation platform: Case study on the Ningbo–Zhoushan port," *Int. J. Inf. Manage.*, vol. 57, Apr. 2021, Art. no. 102133, doi: 10.1016/j.ijinfomgt.2020.102133.

[18] B. Xie, J. Chen, and L. Shen, "Cooperation algorithms in multi-agent systems for dynamic task allocation: A brief overview," in *Proc. 37th Chin. Control Conf. (CCC)*, Jul. 2018, pp. 6776–6781, doi: 10.23919/ChiCC.2018.8483939.

[19] H. Yedidsion, R. Zivan, and A. Farinelli, "Applying max-sum to teams of mobile sensing agents," *Eng. Appl. Artif. Intell.*, vol. 71, pp. 87–99, May 2018, doi: 10.1016/j.engappai.2018.02.017.

[20] L. Vig and J. A. Adams, "Market-based multi-robot coalition formation," in *Distributed Autonomous Robotic Systems 7*, M. Gini and R. Voyles, Eds. Tokyo, Japan: Springer, 2006, pp. 227–236.

[21] N. Nedjah, R. M. D. Mendonça, and L. D. M. Mourelle, "PSO-based distributed algorithm for dynamic task allocation in a robotic swarm," *Proc. Comput. Sci.*, vol. 51, pp. 326–335, Jan. 2015, doi: 10.1016/j.procs.2015.05.250.

[22] W. Xiang and H. P. Lee, "Ant colony intelligence in multi-agent dynamic manufacturing scheduling," *Eng. Appl. Artif. Intell.*, vol. 21, no. 1, pp. 73–85, Feb. 2008, doi: 10.1016/j.engappai.2007.03.008.

[23] Q. Hou and J. Dong, "Robust adaptive event-triggered fault-tolerant consensus control of multiagent systems with a positive minimum interevent time," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 7, pp. 4003–4014, Jul. 2023, doi: 10.1109/TSMC.2023.3238709.

[24] Q. Hou and J. Dong, "Cooperative fault-tolerant output regulation of linear heterogeneous multiagent systems via an adaptive dynamic event-triggered mechanism," *IEEE Trans. Cybern.*, vol. 53, no. 8, pp. 5299–5310, Aug. 2023, doi: 10.1109/TCYB.2022.3204119.

[25] J. C. Amorim, V. Alves, and E. P. de Freitas, "Assessing a swarm-GAP based solution for the task allocation problem in dynamic scenarios," *Expert Syst. Appl.*, vol. 152, Aug. 2020, Art. no. 113437, doi: 10.1016/j.eswa.2020.113437.

[26] R. M. Zlot, "An auction-based approach to complex task allocation for multirobot teams," Ph.D. dissertation, Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, 2006.

[27] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 939–954, Sep. 2004, doi: 10.1177/0278364904045564.

[28] E. Nunes, M. Manner, H. Mitiche, and M. Gini, "A taxonomy for task allocation problems with temporal and ordering constraints," *Robot. Auto. Syst.*, vol. 90, pp. 55–70, Apr. 2017, doi: 10.1016/j.robot.2016.10.008.

[29] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *Int. J. Robot. Res.*, vol. 32, no. 12, pp. 1495–1512, Oct. 2013, doi: 10.1177/0278364913496484.

[30] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, May 1996, doi: 10.1613/jair.301.

[31] N. Jiang, Y. Deng, A. Nallanathan, and J. A. Chambers, "Reinforcement learning for real-time optimization in NB-IoT networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1424–1440, Jun. 2019, doi: 10.1109/JSAC.2019.2904366.

[32] S. Adam, L. Busoniu, and R. Babuska, "Experience replay for real-time reinforcement learning control," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 2, pp. 201–212, Mar. 2012, doi: 10.1109/TSMCC.2011.2106494.

[33] X. Ma, H. Xu, H. Gao, M. Bian, and W. Hussain, "Real-time virtual machine scheduling in industry IoT network: A reinforcement learning method," *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 2129–2139, Feb. 2023, doi: 10.1109/TII.2022.3211622.

[34] N. D. Daw and P. N. Tobler, "Value learning through reinforcement: The basics of dopamine and reinforcement learning," in *Neuroeconomics*, 2nd ed., P. W. Glimcher and E. Fehr, Eds. New York, NY, USA: Academic, 2014, ch. 15, pp. 283–298, doi: 10.1016/B978-0-12-416008-8.00015-2.

[35] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[36] X. Zhao, Q. Zong, B. Tian, B. Zhang, and M. You, "Fast task allocation for heterogeneous unmanned aerial vehicles through reinforcement learning," *Aerosp. Sci. Technol.*, vol. 92, pp. 588–594, Sep. 2019, doi: 10.1016/j.ast.2019.06.024.

[37] R. J. Alitappeh and K. Jeddisaravi, "Multi-robot exploration in task allocation problem," *Appl. Intell.*, vol. 52, no. 2, pp. 2189–2211, Jan. 2022, doi: 10.1007/s10489-021-02483-3.

[38] K. Tuyls and G. Weiss, "Multiagent learning: Basics, challenges, and prospects," *AI Mag.*, vol. 33, no. 3, pp. 41–52, Sep. 2012, doi: 10.1609/aimag.v33i3.2426.

[39] Y. Song, H. Zhao, R. Luo, L. Huang, Y. Zhang, and R. Su, "A SUMO framework for deep reinforcement learning experiments solving electric vehicle charging dispatching problem," 2022, *arXiv:2209.02921*.

[40] Y. Liu, F. Wu, C. Lyu, S. Li, J. Ye, and X. Qu, "Deep dispatching: A deep reinforcement learning approach for vehicle dispatching on online ride-hailing platform," *Transp. Res. E, Logistics Transp. Rev.*, vol. 161, May 2022, Art. no. 102694, doi: 10.1016/j.tre.2022.102694.

[41] Z. Liu, J. Li, and K. Wu, "Context-aware taxi dispatching at city-scale using deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 1996–2009, Mar. 2022, doi: 10.1109/TITS.2020.3030252.

[42] Y. Jiao, X. Tang, Z. Qin, S. Li, F. Zhang, H. Zhu, and J. Ye, "Real-world ride-hailing vehicle repositioning using deep reinforcement learning," *Transp. Res. C, Emerg. Technol.*, vol. 130, Sep. 2021, Art. no. 103289, doi: 10.1016/j.trc.2021.103289.

[43] H. Kumar, A. Koppel, and A. Ribeiro, "On the sample complexity of actor-critic method for reinforcement learning with function approximation," *Mach. Learn.*, vol. 112, no. 7, pp. 2433–2467, Jul. 2023, doi: 10.1007/s10994-023-06303-2.

[44] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, vol. 48, 2016, pp. 1928–1937. [Online]. Available: https://proceedings.mlr.press/v48/mniha16.html

[45] D. Fleckenstein, R. Klein, and C. Steinhardt, "Recent advances in integrating demand management and vehicle routing: A methodological review," *Eur. J. Oper. Res.*, vol. 306, no. 2, pp. 499–518, Apr. 2023, doi: 10.1016/j.ejor.2022.04.032.

[46] N. Wang and J. Guo, "Multi-task dispatch of shared autonomous electric vehicles for mobility-on-demand services—Combination of deep reinforcement learning and combinatorial optimization method," *Heliyon*, vol. 8, no. 11, Nov. 2022, Art. no. e11319, doi: 10.1016/j.heliyon.2022.e11319.

[47] L. Canese, G. C. Cardarilli, L. Di Nunzio, R. Fazzolari, D. Giardino, M. Re, and S. Spanò, "Multi-agent reinforcement learning: A review of challenges and applications," *Appl. Sci.*, vol. 11, no. 11, p. 4948, May 2021, doi: 10.3390/app11114948.

[48] K. Menda, Y.-C. Chen, J. Grana, J. W. Bono, B. D. Tracey, M. J. Kochenderfer, and D. Wolpert, "Deep reinforcement learning for event-driven multi-agent decision processes," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 4, pp. 1259–1268, Apr. 2019, doi: 10.1109/TITS.2018.2848264.

[49] T. Kurczveil, P. López, and E. Schnieder, "Implementation of an energy model and a charging infrastructure in SUMO," in *Simulation of Urban Mobility* (Lecture Notes in Computer Science), vol. 8594, M. Behrisch, D. Krajzewicz, and M. Weber, Eds. Berlin, Germany: Springer, 2013, doi: 10.1007/978-3-662-45079-6_3.

[50] G. H. Bham and R. F. Benekohal, "A high fidelity traffic simulation model based on cellular automata and car-following concepts," *Transp. Res. C, Emerg. Technol.*, vol. 12, no. 1, pp. 1–32, Feb. 2004, doi: 10.1016/j.trc.2002.05.001.

[51] S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges," *Proc. IEEE*, vol. 107, no. 8, pp. 1697–1716, Aug. 2019, doi: 10.1109/JPROC.2019.2915983.

[52] E. W. Dijkstra, "A note on two problems in connexion with graphs," in *Edsger Wybe Dijkstra: His Life, Work, and Legacy*, 1st ed. New York, NY, USA: Association for Computing Machinery, 2022, pp. 287–290.

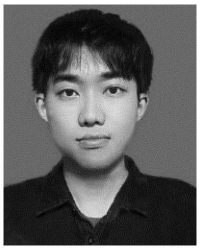
[53] Z. Yi and J. Smart, "A framework for integrated dispatching and charging management of an autonomous electric vehicle ride-hailing fleet," *Transp. Res. D, Transp. Environ.*, vol. 95, Jun. 2021, Art. no. 102822, doi: 10.1016/j.trd.2021.102822.

[54] M. van der Heijden, M. Ebben, N. Gademann, and A. van Harten, "Scheduling vehicles in automated transportation systems," *OR Spectr.*, vol. 24, no. 1, pp. 31–58, Feb. 2002, doi: 10.1007/s291-002-8199-x.

[55] Y. Luo, T. Zhu, S. Wan, S. Zhang, and K. Li, "Optimal charging scheduling for large-scale EV (electric vehicle) deployment based on the interaction of the smart-grid and intelligent-transport systems," *Energy*, vol. 97, pp. 359–368, Feb. 2016, doi: 10.1016/j.energy.2015.12.140.

[56] S. Pateria, B. Subagdja, A.-H. Tan, and C. Quek, "Hierarchical reinforcement learning: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1–35, Jun. 2022, doi: 10.1145/3453160.

SHUPEI ZHANG received the bachelor's, master's, and Ph.D. degrees in vehicle engineering from the School of Automotive Engineering, Jilin University, China. In 2011, he joined the School of Automotive and Traffic Engineering, Jiangsu University, where he is currently an Associate Professor. His research interests include continuously variable transmission technology and automotive testing techniques. He has published numerous research articles in his field, including studies on vehicle steady-state control, analysis of regenerative braking systems, and energy recovery characteristics of electric vehicles.



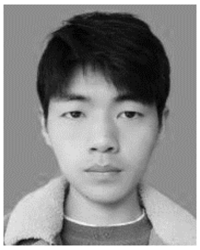
HUAPENG SHI received the B.S. degree from Jiangsu University, Zhenjiang, China, where he is currently pursuing the Ph.D. degree.

His research interests include vehicle-road collaboration, reinforcement learning, and vehicle decision-making.

WEI ZHANG received the B.S. degree in computer application technology and the M.S. and Ph.D. degrees in traffic information engineering and control from Jilin University, Changchun, China.

In 2011, she joined the School of Automotive and Traffic Engineering, Jiangsu University, where she is currently a Lecturer. She has made significant contributions in the domain of traffic information collection and processing technology.

YING PANG received the B.S. degree from Jiangsu University, Zhenjiang, China, where she is currently pursuing the master's degree. Her research interests include deep learning, intelligent transportation systems, and intelligent vehicles.



PENGJU SUN is currently pursuing the degree in vehicle engineering with Jiangsu University, Zhenjiang, China. His research interests include reinforcement learning and vehicle decision-making.