Online Parking Assignment in an Environment of Partially Connected Vehicles: A Multi-Agent Deep Reinforcement Learning Approach

Citation for published version (APA):
Zhang, X., Zhao, C., Liao, F., Li, X., & Du, Y. (2022). Online Parking Assignment in an Environment of Partially Connected Vehicles: A Multi-Agent Deep Reinforcement Learning Approach. *Transportation Research. Part C: Emerging Technologies*, *138*, Article 103624. https://doi.org/10.1016/j.trc.2022.103624

Document license:
TAVERNE

DOI:
10.1016/j.trc.2022.103624

Document status and date:
Published: 01/05/2022

Document Version:
Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Transportation Research Part C 138 (2022) 103624

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Online parking assignment in an environment of partially connected vehicles: A multi-agent deep reinforcement learning approach

Xinyuan Zhang [a], Cong Zhao [b,*], Feixiong Liao [c], Xinghua Li [a,b], Yuchuan Du [b]

[a] Urban Mobility Institute, Tongji University, Shanghai 200092, China
[b] Key Laboratory of Road and Traffic Engineering of the Ministry of Education, Tongji University, Shanghai 201804, China
[c] Urban Planning and Transportation Group, Eindhoven University of Technology, the Netherlands

A R T I C L E I N F O

Keywords:
Online parking assignment
Partially connected vehicles
Multi-agent
Deep reinforcement learning

A B S T R A C T

The advent of connected vehicles (CVs) provides new opportunities to address urban parking issues due to the widespread application of online parking assignment (OPA) services. However, before CVs fully replace non-connected vehicles (NCVs), it is envisioned that CVs and NCVs will coexist for a long time. This brings challenges for OPA because of the availability constraints imposed by the uncertain arrivals and departures of NCVs. This paper proposes a multi-agent deep reinforcement learning framework to generate efficient OPA strategies with partial observations of parking demand. Specifically, we create two agents, one for measuring the impact of NCVs and the other for exploring the parking characteristics of CVs. A value decomposition method is adopted to solve the multi-agent learning problem, and a modified exploration strategy is designed to direct agent training and avoid unnecessary trials. To verify the performance of the proposed approach, we derive the baselines of the total time expenditure in a parking area based on the widely adopted first-come-first-served strategy and a hypothetical system optimum strategy, respectively. Also, we present a dynamic assignment model with forecasting as a comparison of the proposed approach with the same demand information. Two typical parking scenarios are selected to conduct comparative experiments with actual operating data. The experimental results show that the proposed learning-based approach can effectively allocate parking resources. Provided with user parking information of CVs short in advance, our approach can achieve up to 15% improvement in assignment performance compared with other baselines.

1. Introduction

Finding vacant parking spots is often a headache for car drivers in most large cities. The corresponding cruising-for-parking traffic has a significant influence on congestion and emission (Zhao et al., 2021; Zhao et al., 2022). It is revealed that cruising-for-parking vehicles account for over 8% of traffic in 11 major cities, and over 8 min are spent to find parking spots in rush hours (Shoup, 2006). For example, it was found that 15% of traffic searches for available parking spots in the district of Stuttgart, Germany (Hampshire and Shoup, 2018). Meanwhile, it is shown that 63 million vehicle miles traveled (VMT) are generated for cruising-for-parking every year in Chicago, USA, which would account for a waste of over 3.1 million gallons of gasoline and over 48,000 tons

* Corresponding author.
E-mail address: zhc@tongji.edu.cn (C. Zhao).

https://doi.org/10.1016/j.trc.2022.103624
Received 2 August 2021; Received in revised form 30 January 2022; Accepted 23 February 2022
Available online 4 March 2022

of carbon dioxide emissions (Ayala et al., 2011).

To reduce drivers' cruising time, researchers have investigated various parking demand management strategies (Zhao et al., 2018), such as parking pricing, tradable parking permits, parking time limits, etc. For example, Qian and Rajagopal (2015) proposed a scheme of dynamic parking pricing for recurrent parking demand of morning commuting, which essentially reduced cruising-for-parking traffic. The famous program in this direction is *SFpark*, which supports curbside parking management in San Francisco through a large-scale dynamic parking pricing scheme (Chatman and Manville, 2014). *SFpark* collects and distributes real-time parking information and dynamic prices to users to reduce cruising times. However, parking pricing is not an effective way when the parking supplies are inadequate at the trip destinations, which would lead to intensive competition. Wang et al. (2020) investigated the hybrid management scheme of optimal parking pricing and parking permits, which is proved to be more effective than pure parking pricing or pure parking permits. In addition, these parking demand management strategies need to be combined with information provision to play a greater effect (Qian and Rajagopal, 2015; Lei and Ouyang, 2017).

With the rapid development of emerging information and communication technologies, a number of intelligent parking systems have been developed to optimize urban parking operation and management. For example, parking guidance and information (PGI) systems have been developed in most large cities over two decades (Thompson et al., 2001). The location and occupancy information of nearby parking lots is displayed on variable message signs (VMS) at major streets and intersections. However, these parking information provision systems may easily guide crowds of vehicles to the same parking facility simultaneously, which inevitably causes local congestion and fierce competition, and consequently lowers service quality (Geng and Cassandras, 2013). Additionally, the widespread adoption of smartphones and connected vehicles (CVs) makes it easy for drivers to access such parking information (Zhao et al., 2019). The development of parking assignment and navigation systems receives increasing momentum.

The parking assignment problems can be classified into short-horizon and long-horizon planning problems, where the main difference lies in the relative information availability (e.g., parking arrivals and departures) in advance. Many models and algorithms have been developed to solve the short-horizon planning problem (Ayala et al., 2011; Mladenović et al., 2020; Shao et al., 2016); however, new requests or other unforeseeable events may occur, rendering these short-horizon models inapplicable to real-world cases. To fill this gap, the assignment problem in the long-horizon version is presented by repeatedly tackling the user-resource matching problem and examining earlier decisions with newly available information. Thus, solving these problems in the long-horizon is much more complex than the short version. The challenge lies in making a series of assignment decisions with long-term perspectives to meet the overall objective under incomplete or partial observable environment settings. Specifically, problems that are sequentially solved as soon as new information is revealed, are also known as online problems. online parking assignment (OPA) saves the time of users waiting for an allocation and becomes the focus of this research area.

Thus far, there has been no unified framework and solution for OPA in realistic settings (Mladenović et al., 2021). In previous research, two common assumptions limit the large-scale application of OPA systems: i) all users report the relative parking information (e.g., real-time location, parking preference, and parking reservation) to the operator, and then ii) the operator assigns a suitable parking spot to each user. As a result, the existing OPA methods are effective only if all users' information is known, that is, all vehicles are CVs, or users have other intelligent terminals to communicate with the operator. It can be expected that CVs and non-connected vehicles (NCVs) will coexist for a long time before CVs fully replace NCVs (Jiang et al., 2017), which brings great challenges for urban parking management. Moreover, the OPA problem is demanding in terms of solution time in a real application, which puts forward new requirements for the solution algorithm.

The recent progress in deep reinforcement learning (DRL) has shed light on solving complex sequential decision-making problems in many domains (Mnih et al., 2015). DRL combines reinforcement learning (RL) with deep neural networks (DNNs) to create agents that can act intelligently in complex situations. Among them, RL can achieve better generalization capability as the agents learn decision-making mechanisms from data rather than parameter estimation through fitting the data (Sutton and Barto, 2018). DNNs are adapted at general function approximators that can achieve higher accuracy in approximating the complicated relationship between observations and actions. Moreover, DRL has shown remarkable success in various transportation problems in recent years, including traffic signal control (Li et al., 2021), ride sourcing systems (Mao et al., 2020; Tang et al., 2020), public transport management (Jiang et al., 2018; Wang and Sun, 2020), and intelligent control of autonomous vehicles (Du et al., 2022; Chen et al., 2019a).

DRL provides a feasible approach to solve the OPA problem in a mixed environment of CVs and NCVs. For example, Wang et al. (2021) and Zhang et al. (2021) proposed DRL methods to solve OPA problems in a centralized way for automated multistory parking facilities and automated valet parking. However, they both assume by default that all vehicles are controllable by the operator, which is not applicable in general situations. Applying the single-agent paradigm directly to a mixed environment makes it difficult to distinguish the impacts between NCVs and the external environment. The highly dynamic and stochastic environment prevents the agent from converging to a good assignment strategy. In addition, the DRL methods mentioned above do not consider the long-term effects of parking duration variations. High-graded parking resources may be assigned to users who park for a long period, leaving other users to take a parking lot with a long walking distance. The superposition of these imperfect decisions at an early stage would quickly make the agents fall into a local optimum and prevent agents from exploration for better strategies.

To address the above limitations, a promising solution is first to learn and quantify the impact of NCVs and remove it from the external environment, and thus generate a stationary experience for the agent. Then, a new agent is created to assign the NCVs to transfer their impacts and achieve cooperative decision-making between the two agents. Moreover, an exploration strategy that considers the variation of users' parking duration could avoid a local optimum in long-term assignment tasks. Therefore, this study proposes a multi-agent deep reinforcement learning (MARL) framework with a modified exploration strategy to solve the OPA problem in a mixed environment of CVs and NCVs. The framework allows us to reduce the total travel time of both CVs and NCVs by only observing and dispatching CVs to appropriate parking lots. Based on a series of simulations and comparisons with other baseline methods, results show that the MARL framework can effectively allocate parking resources under different CV penetration rates. Our main contributions are as follows:

i) To deal with the highly dynamic and stochastic environment when implementing the original single-agent paradigm to solve the OPA problem, we add a new agent for quantifying the impact of NCVs, thus forming a MARL framework, and propose appropriately designed DRL formulations.
ii) To facilitate the exploration of agents in reliable directions, we design an efficient exploration strategy combined with the proposed MARL framework, which can avoid falling into a local optimum in long-term assignment due to random selections at an early stage.
iii) To demonstrate the superiority of the proposed framework, we first formulate several rule-based and model-based baselines, and then design a real-world parking simulator for quantitative studies and comparisons.

The remainder of this paper is organized as follows. We first review the related literature in Section 2. In Section 3, we describe the OPA problem in a mixed environment of CVs and NCVs, and provide typical baselines for comparisons. Section 4 describes the proposed MARL framework, including the DRL formulations, learning algorithm, and exploration strategies. Section 5 presents our experiments and simulation results, and Section 6 concludes our work and suggests future research directions.

2. Related work

According to the literature, parking assignment services are often formulated as a dynamic resource assignment (DRA) problem. Mouskos et al. (2000) proposed a static and deterministic parking reservation system to address parking problems in major metropolitan areas, in which the objective is to minimize the system-wide parking cost based on binary integer linear programming. Geng and Cassandras (2013) designed an OPA system using a queueing model and formulated the problem with mixed-integer linear programming (MILP), which can efficiently reduce drivers' cruising time and parking cost. Based on this, Kotb et al. (2016) extended their work by dividing users into multiple types and adding a dynamic pricing mechanism to maximize the utilization of parking resources. Chen et al. (2015) developed a smartphone-based parking reservation system to address downtown curbside parking issues. They applied the Vickrey-Clark-Groves mechanism to ensure users truthfully report their destinations. Zhao et al. (2019) proposed an OPA system for mixed human-driven and automated vehicles, where the problem was also formulated with MILP, and the aim was to minimize the total cost at each decision point. However, these studies simplify the parking behavior to some extent.

Furthermore, some studies investigated users' parking behaviors in an OPA system. Ayala et al. (2011) developed a game-theoretic framework by considering multi-user competitions. It is demonstrated that the problem can be regarded as a Nash equilibrium under the complete information condition, which is equivalent to the stable marriage problem (SMP) (Chen et al., 2019b). He et al. (2015) extended the formulation of the SMP with a nonlinear equation system and explored ways of steering the system from user equilibrium to system optimum based on optimal pricing schemes. Chen et al. (2019b) applied the two-sided matching theory to develop a practical parking assignment and navigation system, of which a distributed stable matching algorithm is developed to search the user-optimal solutions without disclosing the privacy. Du and Gong (2016) modeled the local parking competition process using the stochastic Poisson game and designed a decentralized and coordinated mechanism to guide users to parking facilities with the consideration of their preferences. However, the aforementioned studies commonly assume that users can only park based on reservations or allocations from the management center, where the full information is available. The system allocates the current most appropriate parking slots to users by minimizing a cost-related objective function at each decision point in a deterministic environment. The fact that NCVs often arrive at the parking lots randomly and uncontrollably is not taken into account, and thus invalidates the assumption of complete information.

A few studies have addressed uncertain demand and supply in parking assignment problems. Ayala et al. (2011) developed a gravity-based parking algorithm to efficiently assign parking spots to users in a context of incomplete information. Zou et al. (2015) designed a dynamic incentive mechanism to collect drivers' information for improving their activeness in participation and maximizing social welfare. To address the issues of stochastic parking arrivals and departures, Zargayouna et al. (2016) proposed a multi-agent framework with two cooperative models to solve the OPA problem. They found that the fully cooperative agents are better for reducing the time spent on cruising-for-parking. Zhang et al. (2018) designed a shared parking platform to allocate parking resources to two types of users (reservation-based and walk-in) and verified the relationship between reservation limit and the expected revenue. Although the aforementioned model-based studies have shown promising applicability, the centralized OPA models and algorithms require heavy computations, causing limitations in the numbers of users and resources for responsive parking management.

The existing approaches typically solve the optimal allocation based on the current state information but overlook the long-term effects of intermediate parking assignment decisions. To solve this problem, some studies began to address the OPA problem with the advantage of the RL approach. For example, Wang et al. (2021) proposed a two-stage, online operations approach for a multistory parking facility to achieve long-term benefits. The first stage uses an RL approach to determine the parking level selections for incoming demands, while the second stage builds on the first stage results with mathematical programming to optimize the action sequences of the automated elevator. A similar model was proposed in Zhang et al. (2021), which uses an RL approach to choose parking zone and guides automatic guided vehicle (AGV) to avoid collision and deadlock in the conflict zone during the driving process.

Moreover, the problem of the OPA shares some similar properties with the problem of taxi order dispatching, where RL methods have been widely applied. Lin et al. (2018) focused on the large-scale fleet management problem for ride-sharing platforms via a contextual MARL approach to deal with the numerous homogeneous agents. Also, Shou and Di (2020) applied the mean-field theory to effectively achieve large-scale taxi fleet management. Although the complexity of the problem can be significantly reduced by treating each taxi as an agent, the decentralized dispatching strategy may sacrifice system-level optimality. Therefore, Mao et al. (2020) developed a centralized order dispatching method to maximize the system-level returns. Tang et al. (2020) proposed an advisor-student RL framework to solve the online operation problem of the autonomous electric taxi fleet. The advisor uses the DRL method to determine the number of taxis to serve travel demand, be dispatched, and be recharged in each area. The student follows up a combinational optimization model to precisely assign each taxi to a specific user. Wang et al. (2021), Zhang et al. (2021) also followed this two-stage optimization framework. Moreover, some studies developed MARL frameworks to learn to delay the matching process of ride-sourcing systems (Ke et al., 2020; Qin et al., 2021). The results show that an appropriate delayed matching can accumulate sufficient pairs of idle drivers and unserved passengers, leading to better matching results.

Due to the similarity to the taxi order dispatching problem, one would think that the DRL method would straightforwardly offer effective solutions for the OPA problem. However, the OPA problem and the taxi order dispatching problem have a major difference in the aftermath of intermediate decisions. Specifically, an improper order dispatching may affect a taxi only for a short period because each new dispatching decision erases the influence of the previous one. In other words, the taxi always has a chance to start over and eliminate all previous impacts regardless of its current condition to reach an ideal state. On the other hand, the OPA problem does not have such a feature. Once a CV is assigned to a parking lot, this assignment affects the following decisions, and the influence lasts until

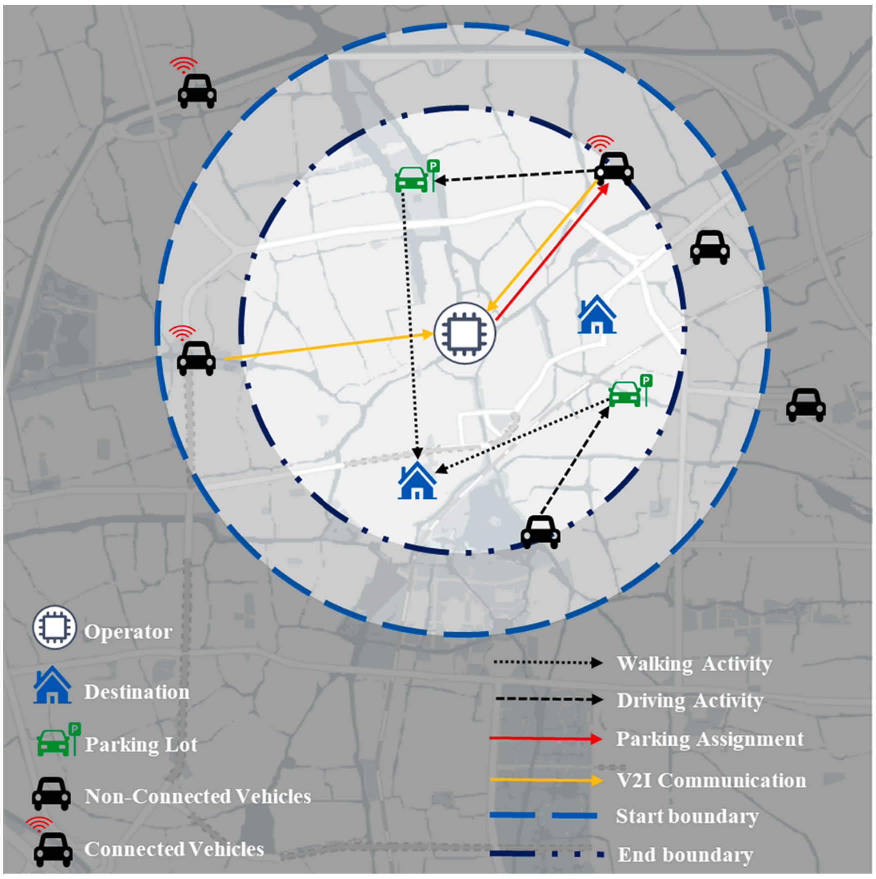


Fig. 1. Representation of the OPA problem in a mixed environment of CVs and NCVs.

the vehicle leaves the parking lot. Therefore, each decision needs to be made tactfully, especially for the CVs with long parking duration. The assignment strategies of agents are derived from trial and error, meaning that it is unlikely for agents to avoid imperfect decisions during the learning process. The superposition of these imperfect decisions would quickly make the agents fall into a local optimum and prevent agents from exploration for better strategies.

In brief, the existing RL-based methods of taxi order dispatching often set up a novel algorithm for a specific scenario, and cannot be straightforwardly applied in our research problem. More importantly, the methods assume that the taxis belonging to the platform are fully visible and controllable, which is the same assumption as that in existing OPA studies and thus is inapplicable to a mixed environment of CVs and NCVs.

3. Problem statement and baselines

In this section, we specify the OPA problem and provide the upper and lower bound formulations, which will be used as the comparison benchmarks for our approach.

3.1. Problem statement

The problem concerns a relatively sealed area $\mathscr{G}$ with multiple heterogeneous parking lots and user destinations. Let $p \in \{1, 2, \ldots, \overline{p}\}$ and $i \in \{1, 2, \ldots, \overline{i}\}$ be the indices of parking lots and destinations, respectively. On the demand side, each user $c \in \{1, 2, \ldots, \overline{c}\}$ needs to reach him/her destination and find an available parking spot. Beyond that, there are two types of users in $\mathscr{G}$: CVs and NCVs. CVs are loaded with vehicle to infrastructure (V2I) (Jiang et al., 2021) modules to enable bi-directional communications with only one operator of $\mathscr{G}$. NCVs are conventional vehicles and do not have V2I modules for communicating with the operator. We assume that the users of CVs and NCVs have the same distributions of arrival time and parking duration. The service time is represented by discrete time intervals $\{1, 2, \ldots, \overline{t}\}$ with time length Δt as one unit of time interval.

As shown in Fig. 1, there are two boundaries in $\mathscr{G}$, namely, start boundary and end boundary. It is assumed that the operator can acquire CVs' real-time location via V2I and move them into the assignment pool when they drive into the start boundary. This indicates that the CVs are close to their destination and have an opportunity to be assigned to appropriate parking lots. Then, CVs share their travel information, including the destination c^i and expected parking duration c^d. The operator needs to accomplish the parking assignment task for the CVs before they arrive at the end boundary. All CVs between the start and end boundaries are in the parking assignment pool, this is in line with the reality of V2I techniques (Jiang et al., 2021) and similar to the matching pool of ride-sourcing systems (Ke et al., 2020). After receiving assigned parking lots c^p, CVs comply with the assignments, drive to the corresponding parking lots, and find free parking spots by themselves. After parking, the drivers walk to their destinations.

The cruising time η_t^p for finding an available parking spot on a parking lot is related to its occupancy (Levy et al., 2013) as

$$\eta_t^p = e_p \left(1 - \frac{\delta_t^p}{\sigma^p}\right)^{-1} \quad \forall t, p \tag{1}$$

where e_p represents the average cruising time for empty parking lot p; δ_t^p and σ^p indicate the current and maximum occupancies of parking lot p, respectively. The walking time from parking lot p to destination i is considered as a constant for simplicity, denoted by w^{pi}. We suppose the total travel time of a user is composed of cruising time and walking time.

The assumption that all CVs comply with the assignments seems unrealistic because the parking lots assigned by the operator may be different from the parking lots that the users want to choose. However, with this assumption, we can significantly simplify the formulations of the DRL model; to remedy this, we will weaken this assumption by setting incomplete obedience rates of CVs to test the model's performance in the experimental section.

For NCVs, the operator does not assign any parking lots due to the lack of communication and even cannot observe them coming. On this basis, we suppose that each NCV finds an optimal parking lot according to the rule of minimizing its own total travel time at the time interval of arriving at the parking area. Other than that, NCVs have the same settings as CVs.

To improve the attractiveness of the parking area, the operator aims to minimize the sum of travel time of all users of CVs and NCVs. Note that, for simplicity, we do not consider parking prices and the cases where the assigned parking lots are fully occupied when CV arrives. Specifically, at time t, the operator observes upcoming CVs and receives their parking-related information for a future period τ, as shown by the yellow line in Fig. 1. Subsequently, the operator dynamically assigns a parking lot to each CV when it arrives at the end boundary. At the same time, the unobservable and uncontrollable NCVs are randomly mixed with CVs, affecting the assignments of the operator. Therefore, this study explores the OPA problem in a scenario with partially observable and controlled users.

3.2. Baselines

This subsection introduces three typical baselines of parking assignment, which are based on the first-come-first-served (FCFS) strategy, the system optimum (SO) strategy based on complete demand information (Shao et al., 2016), and dynamic assignment (DA) with demand prediction (Geng and Cassandras, 2013), respectively. The solutions produced by the FCFS and SO strategies correspond to the upper and lower bounds of the proposed approach (upper bound corresponds to the numerical cap of the objective function and vice versa for the lower bound), and DA is regarded as a fair comparison method.

3.2.1. *First-come-first-served (FCFS)*

The FCFS assignment applies a myopic strategy that seeks to minimize individual-level waiting time. When a user arrives at the end boundary, the operator allocates a parking spot with the minimum travel time. It has the same choice behavior as NCV users according to the settings in Section 2.1. The FCFS strategy is a special case of dynamic programming and only requires the current environment information. Therefore, we take FCFS as the optimal upper bound of the OPA problem; in other words, any effective parking assignment method should be better than the FCFS strategy, which can be described briefly as

$$c^p = \mathrm{argmin}_p \left(\eta_t^p + w^{pi}\right) \quad \forall p \tag{2}$$

3.2.2. *System optimum (SO)*

The optimal lower bound can be obtained from a binary integer linear programming problem under an imaginative condition that global parking information is gathered beforehand (for example, one day earlier). Since the operator is informed of the complete parking information before the assignments are made, the SO belongs to the static problem and has sufficient horizon and potential to find the optimal solution for the assignments.

Given $\overline{q}$ parking spots, we introduce a binary indicator s_t^q, which equals 1 if parking spot $q \in \{1, 2, \ldots, \overline{q}\}$ is available at time interval t and 0 otherwise. Similarly, let $\overline{c}$ be the total number of users (parking demand) and r_t^c is another binary indicator, which equals 1 if user c coincides with time interval t and 0 otherwise. We further introduce a binary decision variable x^{cq}: $x^{cq} = 1$ if user c is assigned to parking spot q and $x^{cq} = 0$ otherwise. Based on the above definitions, we use a binary indicator z_t^q defined below to represent parking occupancy of q:

$$z_t^q = \sum_c x^{cq} r_t^c \quad \forall t, q \tag{3}$$

where z_t^q indicates the status of parking spot q at time interval t: 1 is occupied and 0 otherwise. The operator aims to minimize the total travel time respecting constraints (1) and (5)-(8):

$$\min \sum_t \sum_c \sum_q x^{cq} d^{qp} \widetilde{r}_t^c \left(\eta_t^p + w^{pi}\right) \tag{4}$$

subject to

$$\sum_q x^{cq} = 1 \quad \forall c \tag{5}$$

$$z_t^q \leqslant s_t^q \quad \forall t, q \tag{6}$$

$$\eta_t^p = \sigma^p - \sum_q d^{qp} z_t^q \quad \forall t, p \tag{7}$$

$$x^{cq} \in \{0, 1\} \quad \forall c, q \tag{8}$$

The objective function (4) is to minimize the total travel time, and d^{qp} is a binary indicator equaling 1 if parking spot q belongs to parking lot p and 0 otherwise. t is equal to the time interval when user c arrives at the start boundary, $\widetilde{r}_t^c$ equals 1 and the rest equals 0. Constraint (5) indicates that any user should be assigned to only one available parking spot. Constraint (6) ensures that a parking spot is not be occupied by multiple users at the same time. Constraint (7) determines the available parking spots in each parking lot. Constraint (8) defines x^{cq} as a binary decision variable.

The relationship between the cruising time in the parking lot and the number of available parking spots is nonlinear according to constraint (1) and is correlated with the assignment decisions. Thus, finding the exact solution of the SO becomes intractable. To simplify, we assume that the δ_t^p values are known and deterministic, which is the same assumption as made by Mladenović et al. (2020) in solving the static assignment problem. In other words, we assume that a preliminary statistical analysis has already been done to identify these values at each minute of the day and relax constraint (7). It is worth noting that even if an accurate occupancy estimate is provided, the SO is not guaranteed to yield the optimal solution. For example, if the parking lot p is predicted in advance to be highly occupied, the corresponding required traveling time is also costly, and then the SO avoids assigning users to this parking lot. However, such avoidances lead to increases in the occupancy of other parking lots and a decrease in the occupancy of parking lot p, in turn, making it possible that choice p may be preferable even with the prediction of the high occupancy. Investigating which prediction could yield the optimal solution is beyond the scope of this paper. For simplicity, we use occupancy results derived from other baselines, our proposed RL method, and random assignment strategy and choose the one that performs best.

The remaining constraints in the model are linear, and the decision variables are binary, formulating a binary integer linear programming problem, which is used to solve with some solvers, such as CPLEX. To conclude, the SO relies on the users' arrival times, parking times, destinations, and the number of available spots at each future time intervals and has the potential to reach optimality.

3.2.3. *Dynamic assignment (DA)*

DA is a common approach to solving OPA problems (Geng and Cassandras, 2013; Zhao et al., 2019). To apply DA to solve the OPA problem in a mixed environment, the operator needs to receive the information of all vehicles (CVs and NCVs) between the two boundaries. Thus, a demand prediction technique is required to predict the information of NCVs between the two boundaries accurately. Combining with the prediction technique, DA can handle the OPA problems with partial observations and random parking arrivals and departures. To avoid the influence of prediction error, we assume that the complete information of NCVs within the start boundary and end boundary can be obtained through the demand prediction technique, including users' location, destination, and parking duration. Note that the CVs automatically report their relevant information via the V2I network.

Let $CV(t)$ and $NCV(t)$ be the set of CVs and NCVs in the assignment pool at time interval t, respectively. Let $\psi_p(t)$ be the number of available parking spots on parking lot p at t. We use a binary decision variable $x^{cp}(t)$: $x^{cp}(t) = 1$ if user c is assigned to parking lot p and $x^{cp}(t) = 0$ otherwise.

$$min \sum_{c\in CV(t)\cup NCV(t)} \sum_{p} x^{cp}(t)\left(\eta_t^p + w^{pi}\right) \tag{9}$$

subject to

$$\sum_{p} x^{cp}(t) = 1 \quad \forall c \in CV(t) \tag{10}$$

$$x^{cp}(t) = 1 \quad \forall c \in NCV(t), p = \text{argmin}_p\left(\eta_t^p + w^{pi}\right) \tag{11}$$

$$\sum_{c\in CV(t)\cup NCV(t)} x^{cp}(t) \leqslant \psi_p(t) \quad \forall p \tag{12}$$

$$x^{cp}(t) \in \{0,1\} \quad \forall c,p \tag{13}$$

Objective function (8) aims to find the minimum travel time for all users in the assignment pool. Constraint (9) guarantees that each CV should be assigned to only one parking lot, and constraint (10) indicates that the NCVs follow the FCFS strategy to minimize an individual objective. Constraint (11) ensure that each parking lot cannot accept excessive users. Constraint (12) defines $x^{cp}(t)$ as a binary decision variable. The DA problem is activated by events. When a CV is about to leave the end boundary, the DA assigns parking lots to all users in the assignment pool to ensure that the current assignment for the CV is optimal under the available information. Although DA with demand prediction can handle the problem of partial observation and random parking demand in the mixed environment, the solution is still myopic, and the performance is yet to be tested in long-term assignment tasks.

In summary, reaching the lower bound of assignment performance requires complete parking demand information in advance, while the upper bound can be obtained without any prior knowledge. Thus, it is conceivable that the performance of any method facing partial user information falls between the upper and lower bounds. Then, for comparison, we introduce the general DA to solve the OPA problem in a mixed environment with demand prediction.

4. Methodology

As introduced in Section 1, DRL is in nature a feasible approach to solve the OPA problem, where deep learning can generalize from partial user information to the whole by its powerful learning ability. On the other hand, RL can find an assignment decision that considers the subsequent impacts through trial and error. This section first describes the OPA problem in RL formulations, known as the Markov decision process. Then, the MARL framework and an accompanying exploration strategy are discussed to address the OPA problem.

4.1. *DRL formulation*

The OPA problem is paraphrased into the RL paradigm below using terms of agent, state, action, and reward.

4.1.1. *Agent*

We forge two agents, A^1 and A^2, to assign CVs and NCVs for agent policy learning, respectively. Since NCVs select parking lots based on the principle of minimizing travel time without receiving assignments from the operator, A^2 is unable to assign any NCVs. However, in order to accurately evaluate the impact of NCVs, we indirectly regard that each NCV follows a decision from A^2, the same as choosing a parking lot to minimize travel time. Thus, the impact of NCVs is converted into the impact of the assignment policy of A^2. The policy of A^2 is deterministic, equivalent to the FCFS strategy. On the other hand, the policy of A^1 is uncertain at first and needs to be learned via interacting with the environment. Hence, while A^2 only executes policy evaluation, A^1 executes policy evaluation and iteration successively.

4.1.2. State

The state of the parking assignment should reflect the information of both the management area and users. Assuming that at time t user c arrives at the end boundary, the current state observations cover the area and demand information, entailed as follows.

(a) Area information: Area information observed by A^1 and A^2 is the same. This part of information describes the monitoring indicators of the current situation and the prediction of future parking supply and demand. The current situation of each parking lot at time interval t is denoted as $\left[t, \mu_t^1, \mu_t^2, \ldots, \mu_t^{\bar{P}}\right]$, where μ_t^p represents the intensity of resource utilization of parking lot p as

$$\mu_t^p = \frac{\sum_{q \in p} \phi_t^q}{\sigma^p} \quad \forall t, p \tag{14}$$

where ϕ_t^q represents the remaining parking duration on parking spot q at time t; if spot q is available, then $\phi_t^q = 0$. For CVs, it is straightforward to calculate ϕ_t^q using the provided expected parking duration. For NCVs, we calculate ϕ_t^q based on the average parking time at t in the historical data. Furthermore, the predicted ϕ_t^q is reset immediately when the corresponding vehicle departs in advance. Conversely, if ϕ_t^q counts down to 0 and the vehicle still remains, ϕ_t^q is set to 1.

The prediction of future parking supply is denoted as $\left[b_t^p, b_{t+1}^p, \ldots, b_{t+\tau}^p\right]$, where b_t^p is the number of occupied parking spots released at time t on parking lot p. b_t^p can be derived from ϕ_t^q. For example, if the set of ϕ_t^q on parking lot p has 2 elements equal to 3 at time interval t, it means that 2 spots will be released at time interval $t+3$, the value of b_{t+3}^p is equal to 2. The future parking demand is denoted as $\left[k_t^i + j_t^i, k_{t+1}^i + j_{t+1}^i, \ldots, k_{t+\tau}^i + j_{t+\tau}^i\right]$. k_t^i and j_t^i represent the numbers of CVs and NCVs that arrive at the end boundary at time t and set off to destination i, respectively. k_t^i is obtained automatically through the V2I platform, while the average parking demand at t in the historical data is used to estimate j_t^i. For simplicity, we use the average demand from the historical data multiplied by a predetermined rate of NCVs to predict the demand of NCV users. However, other mathematical and statistical methods or complicated machine learning algorithms are also applicable here.

(b) User information: User information is used to characterize the parking demand, including user destination, parking duration, and user type. User destination is indicated with the One Hot encoding (Ke et al., 2020) for CVs or with the historical selection frequency of each destination at time t for NCVs. Parking duration is provided by CVs or average parking duration at t in the historical data for NCVs. User type is a binary indicator, 1 for CV or NCV, 0 otherwise.

The above information is combined as the environment observations received by the agents. We use o_t^1 and o_t^2 to denote agent observations for A^1 and A^2 at time t, respectively. Furthermore, the One Hot encoding of user type is adopted to replace the last element in agent observations and label this new combination as a true state, denoted as s_t. Compared to the SO and DA baselines, the state does not contain complete information about the parking demands within the start and end boundaries because the RL agent can learn the daily pattern of parking demands by continuously interacting with the environment.

4.1.3. Action

Each agent has the same action space, which refers to the parking lots when a user arrives at the end boundary, denoted as a_t^1 and a_t^2, respectively. It is worth mentioning that a_t^1 is the action made by A^1 after observing o_t^1 according to policy π^1, $a_t^1 = \pi^1\left(o_t^1\right)$ for short, while a_t^2 comes from the FCFS strategy. To avoid assigning a specific user to an over-saturated parking lot and causing additional detours and extra congestion, A^1 screens occupancy of each parking lot before assigning, and only unsaturated parking lots are selected as the output action. At last, we couple actions from A^1 and A^2, and give feedback to the environment together as a collective action, denoted by $\boldsymbol{a}_t$.

4.1.4. Reward function

After taking actions, both agents receive an immediate reward r_t from the environment to assess the collective action. The reward function is based on the intention to minimize the total travel time of all users, defined as:

$$r_t = \frac{E_1 - \left(\eta_t^p + w^{pi}\right)}{E_2} \quad \forall t \tag{15}$$

where E_1 and E_2 are constants to normalize the reward to the range $[0,1]$ as suggested by (Henderson et al., 2018).

4.1.5. State transition

Fig. 2 illustrates the proposed MARL framework. The environment receives the actions from agents and arrives at a new state according to the transition dynamics. When state s_t is generated (a user arrives at t), agents observe the state and take collective action $\boldsymbol{a}_t$, and a user follows the assignment from A^1 or A^2. Then, the environment evaluates this action and returns a reward r_t. After that, the next state $s_{t+\varepsilon}$ (next user arrives at $t+\varepsilon$) is postponed until the subsequent user arrives. At this point, a tuple of state transition $\{s_t, \boldsymbol{a}_t,$

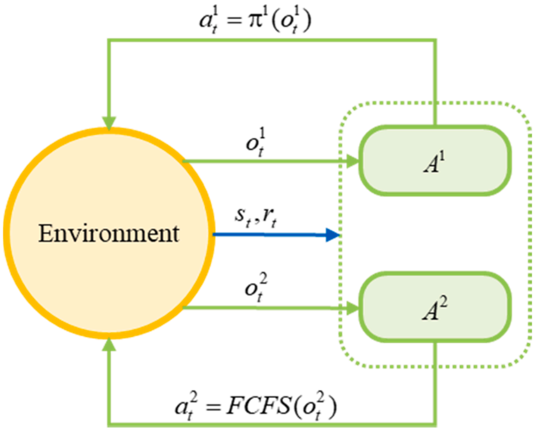


Fig. 2. Schematic overview of the MARL framework.

$r_t, s_{t+\varepsilon}\}$ is formed, and the next step is to use a MARL-based method to discover the relationships between these tuples.

4.2. *Algorithm*

The monotonic value function factorization (QMIX) algorithm combines several deep q-network to solve the problem of unstable training of traditional RL in the MARL framework. We briefly introduce several fundamental RL algorithms in the following subsection before revealing the QMIX. Both q-learning and deep q-network (DQN) are designed for single-agent learning tasks, and for the sake of notational unity, we take the example of agent A^1 here.

4.2.1. *Q-learning and deep q-network (DQN)*

Q-learning is a classical RL algorithm based on the time difference to update $Q_1(o_t^1, a_t^1)$ (Watkins and Dayan, 1992). It computes the discounted sum of rewards earned by performing action a_t^1 in observation (state) o_t^1 and progressively implementing the optimum policy. The agent continuously learns from the environment to update $Q_1(o_t^1, a_t^1)$ via the Bellman optimality equation:

$$Q_1^*(o_t^1, a_t^1) = \mathbb{E}\left[r_t + \gamma \max_a Q_1^*(o_{t+1}^1, a)\right] \tag{16}$$

where γ is the discount factor and ranges from 0 to 1. An iterative update $Q_1^*(o_t^1, a_t^1) \leftarrow \mathbb{E}\left[\mathrm{r_t} + \gamma \max_a Q_1^*(\mathrm{o}_{t+1}^1, \mathrm{a})\right]$ allows Q_1 to converge to Q_1^* with probability 1 as $t \to \infty$. Q-learning uses a table structure to store and update $Q_1(o_t^1, a_t^1)$, and the pressure to store and traverse the table increases dramatically when the dimensionality of the research problem becomes large.

DQN used a deep neural network to improve the representation of the value function in q-learning to accomplish function approximation (Mnih et al., 2015). The above deep neural network's input is the environmental state characteristics, and it will output corresponding value functions of action–state pairs, denoted as $Q_t^1(o_t^1, a_t^1|\theta_1)$. The term θ_1 refers to network parameters. While the agent interacts with the environment, the state transmission pair is recorded as $\{o_t^1, a_t^1, r_t, o_{t+1}^1\}$, and the agent uses these pairs to update $Q_t^1(o_t^1, a_t^1|\theta_1)$ via:

$$Q_1(o_t^1, a_t^1|\theta_1) = Q_1(o_t^1, a_t^1|\theta_1) + \alpha\left(r_t + \gamma max_a Q_1(o_t^1, a|\theta_1) - Q_1(o_t^1, a_t^1|\theta_1)\right) \tag{17}$$

where α is the learning rate of the agent. For optimizing convergence, DQN commonly adopts target q-network output $max_a Q_t^1(o_{t+1}^1, a|\overline{\theta}_1)$ to replace $max_a Q_t^1(o_{t+1}^1, a|\theta_1)$ in Eq. (16), where the current q network parameters update and are copied to the target network periodically or softly. The current q network uses the mean square error as the loss function to update the parameters through backpropagation:

$$\mathscr{L} = \frac{1}{m}\sum_{n=1}^{m}\left(y_n - Q_1(o_n^1, a_n^1|\theta_1)\right)^2 \tag{18}$$

where y_n is the target q value of state transmission pair n equaling $r_n + \gamma max_a Q_1(o_n^1, a|\overline{\theta}_1)$ if o_n^1 is not a terminal observation (state) and r_n otherwise.

DQN has demonstrated exceptional performance in numerous complex problems but tends to be helpless to the agent in the MARL framework for two main reasons. The first reason is that the following observation (state) received by a single agent is unstable and becomes an obstruction of convergence. For example, o_{t+1}^1 depends not only on o_t^1 and a_t^1, but also on a_t^2, which results in the loss of the Markov property for the stored state transmission pair $\{o_t^1, a_t^1, r_t, o_{t+1}^1\}$. The first problem may not be fatal under our DRL formulation since the agent A^2 follows the deterministic assignment strategy, and action a_t^2 can be treated as a part of the environment dynamics. The second reason is that only one immediate reward is returned from the environment as an evaluation of the collective action of all the agents, and it is challenging for each agent to retrieve accurate information from it.

4.2.2. QMIX algorithm

In a multi-agent framework, each agent chooses an action, forming a collective action $\boldsymbol{a}_t$, and shares a global, immediate reward r_t that assesses the collective action taken previously. The collective action has a collective agent-value function $Q_{tot}(s_t, \boldsymbol{a}_t) = \mathbb{E}_{\mathrm{s}_{t+1:\infty}, \mathbf{a}_{t+1:\infty}}[\mathrm{R_t}|\mathrm{s_t}, \mathbf{a_t}]$, where R_t is the discounted return at time t. The main challenge in the MARL framework is to evaluate the contribution of each agent separately and accurately, to obtain individual value function from the collective action-value function. We use $Q_1(o_t^1, a_t^1)$ and $Q_2(o_t^2, a_t^2)$ to represent the individual value functions of agents A^1 and A^2, respectively.

QMIX is a novel value-based technique for training decentralized policies in a centralized end-to-end way (Rashid et al., 2018). To achieve this, QMIX needs to overcome two difficulties. The first one is to measure the influence of each agent's action according to the single global reward in the process of centralized training, as mentioned above. The second difficulty is to ensure that the ensemble of the optimal actions of agents is the optimal action of the ensemble of agents when agents interact in a decentralized manner.

On the first difficulty, QMIX introduces a new component, called the mixing network, and the part called the agent network of the original DQN, as shown in Fig. 3. Agent network is embedded within the agent, taking the observation as input and the individual value function as output. Concerning data fusion processing, each observation portion is individually inputted into processing units, a two-layer fully connected network with a ReLU (rectified linear activation function). The value function is obtained by a fully connected layer that receives the combination of each processing unit's outputs and the preselected action. The agent networks of A^1 and A^2 are not the same, and thus different assignment strategies can be learned. The mixing network is a feed-forward network, which takes the individual value functions as input and mixes them to produce a collective function. The collective action-value function is regarded as a complex nonlinear mixture of each agent's values that solely depend on local observations. Then, the mixing network updates the collective function based on immediate reward r_t and decomposes the collective function to update individual functions.

On the second difficulty, due to the decentralized execution based on centralized training, it is unnecessary to decompose $Q_{tot}(s_t, \boldsymbol{a}_t)$ completely. It is only necessary to ensure that the collective action $\boldsymbol{a}_t$ takes the maximum value in $Q_{tot}(s_t, \boldsymbol{a}_t)$ consisting of $\{a_t^1, a_t^2\}$, which takes the individual maximum value in $\{Q_1(o_t^1, a_t^1), Q_2(o_t^2, a_t^2)\}$, respectively. Specifically, performing global argmax (an operation that finds the argument giving the maximum value) on $Q_{tot}(s_t, \boldsymbol{a}_t)$ yields the same result as a set of individual argmax operations performed on each single value function (Son et al., 2019), which can be expressed as:

$$\mathrm{argmax}_{\boldsymbol{a}_t} Q_{tot}(s_t, \boldsymbol{a}_t) = \begin{pmatrix} \mathrm{argmax}_{a_t} Q_1(o_t^1, a_t) \\ \mathrm{argmax}_{a_t} Q_2(o_t^2, a_t) \end{pmatrix} \tag{19}$$

There are numerous approaches to design the mixing network to satisfy Eq. (18); for example, Value Decompose Network (VDN)

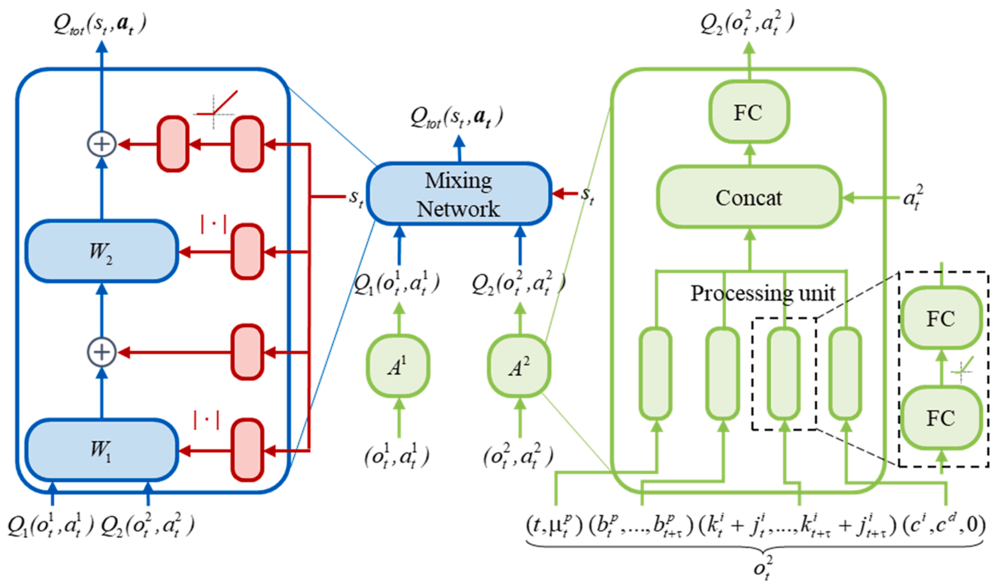


Fig. 3. The framework of the QMIX algorithm.

uses a straightforward method that sums all individual value functions (Sunehag et al., 2018). QMIX takes a step further and finds that designing the mixing network only requires a monotonic relationship between the global value function and each value function, denoted as:

$$\frac{\partial Q_{tot}}{\partial Q_1}, \frac{\partial Q_{tot}}{\partial Q_2} \geqslant 0 \tag{20}$$

To guarantee the monotonicity constraint, QMIX uses separate hyper-networks to generate the weights of the mixing network based on state s_t , and each hyper-network is activated by an absolute function. Except for the final bias created by a two-layer hyper-network with a ReLU, the biases are formed in the same manner but are not limited to being non-negative.

The QMIX algorithm is trained by sampling batches of transitions from the replay memory and minimizing the squared temporal difference error, which is analogous to the standard DQN loss (Mnih et al., 2013).

4.3. Exploration

To optimize the accumulated rewards in RL, an agent must learn how to map states to actions. The probability distribution of the rewards associated with each action, however, is diverse and unknown to the agent. Hence, the agent needs to estimate the distribution of rewards and identify which action to choose to obtain the maximum reward via exploration and exploitation.

Exploitation uses the agent's current action-value estimates to select the greedy action that yields the most reward. Nevertheless, being greedy with action-value estimations may not result in the best outcome and may lead to sub-optimal outcomes. Exploration, on the other hand, allows an agent to increase its current understanding of each action (by selecting a random action), which may result in a long-term gain. Improving the accuracy of anticipated action-values allows a future agent to make more informed judgments. However, it cannot accomplish both simultaneously, which is known as the exploration–exploitation dilemma (Sutton and Barto, 2018).

$\epsilon -$ greedy is a simple, practical, and perhaps the most extensively used approach for balancing exploration and exploitation, where ϵ refers to the likelihood of exploiting most of the time with a small possibility of exploring (Sutton and Barto, 2018). When confronted with long-horizon planning and control problems, such as the OPA problem, the agent would follow a sequential trajectory. $\epsilon -$ greedy might choose inferior but long-term effective actions at random at the early stages. In the OPA problem, limited and high-graded parking resources are vulnerable to be occupied by long-term parking users, assigned by $\epsilon -$ greedy randomly. Long-term parking users would significantly reduce the turnaround of high-graded parking resources, making subsequent users choose farther parking lots, thus increasing travel time.

Therefore, we suggest a modified $\epsilon -$ greedy strategy that considers prior information regarding OPA. We suppose that the following prior knowledge may be optimal for directing an agent exploration and avoiding unnecessary trials.

i) The FCFS strategy makes the near-optimal assignments for the group of users with short parking duration users. The subsequent impact of this assignment is well-confined, and thus pursuing the instantaneous optimum is an efficient way for assignment.
ii) Long-term parking users may be assigned to a parking lot with a long travel time, and high-graded parking resources are reserved for subsequent short-term parking users at an early stage to minimize the total travel time.

Based on the above mechanisms, we set a trend of exploration that the chance for choosing parking lot p by user c is proportional to $e^{-\xi\left|\widetilde{c^d} - \widetilde{T^p}\right|}$, where $\widetilde{c^d}$ is the normalized parking duration within range $[0,1]$, $\widetilde{T^p}$ is the normalized sum of cruising and walking time within range $[0,1]$. ξ is the temperature, which starts from 1 and steadily decreases to 0 at the end. The decay of temperature indicates that the exploration trend continues to decline over time because the subsequent impact of assigning long-term parking users becomes weaker as time goes.

The modified $\epsilon -$ greedy strategy can be obtained by:

$$\begin{cases} c^p = \mathrm{argmax}_p Q_1\left(o_t^1, p\right), \mathrm{with probability} 1-\epsilon \\ P(c^p = p) = \dfrac{e^{-\xi\left|\widetilde{c^d} - \widetilde{T^p}\right|}}{\sum_p e^{-\xi\left|\widetilde{c^d} - \widetilde{T^p}\right|}}, \mathrm{with probability} \epsilon \end{cases} \tag{21}$$

where user c does not belong to the group with short parking duration or use the FCFS strategy.

Exploration determines the learning direction of agents toward unknown states. If an exploration guides agents to the optimal assignment strategy, it will dramatically reduce the training cost. However, the optimal strategy for the OPA problem is unknown in advance. We propose two prior knowledge that may be parts of the optimal assignment strategy and serve as the exploration direction of agents. Compared with the random direction in the original $\epsilon -$ greedy strategy, the modified $\epsilon -$ greedy strategy narrows the state space, which is otherwise explored, and thus accelerates the QMIX algorithm's convergence.

The QMIX algorithm is presented as follows. Line 1 creates a replay buffer to keep state transition. Lines 2–4 initialize the agent network and mixing network of the QMIX algorithm with random parameters, along with the corresponding target network for training. Lines 7–9 correspond to the observations (states), actions, and rewards of the agents, respectively. Lines 14–19 are the standard learning process of the agents.

Algorithm 1. QMIX algorithm for OPA in an environment of partially CVs.

```
1   Initialize replay buffer $\mathcal{D}$ to capacity $\mathcal{C}$
2   Initialize action-value function $Q_1\left(o_t^1, a_t^1 \mid \theta_1\right), Q_2\left(o_t^2, a_t^2 \mid \theta_2\right)$ with random weights $\theta$
3   Initialize target action-value function $Q_1\left(o_t^1, a_t^1 \mid \overline{\theta_1}\right), Q_2\left(o_t^2, a_t^2 \mid \overline{\theta_2}\right)$ with $\overline{\theta} \leftarrow \theta$
4   Initialize mixing network and collective agent-value function $Q_{tot}\left(s_t, \boldsymbol{a_t} \mid \theta\right), Q_{tot}\left(s_t, \boldsymbol{a_t} \mid \overline{\theta}\right)$
5   for $t = 0$ to $\overline{t}$ do
6       if user $c$ arrives at end boundary then
7           Observe $o_t^1, o_t^2, s_t$ and $a_t^2$
8           Sample $a_t^1$ on Eq. (21) and form collective action
9           Receive reward $r_t$ on Eq. (15)
10          if the current user is not the first user then
11              Store tuple $\{\boldsymbol{o_{t-\varepsilon}}, \boldsymbol{a_{t-\varepsilon}}, s_{t-\varepsilon}, r_{t-\varepsilon}, \boldsymbol{o_t}, \boldsymbol{a_t}, s_t\}$ in $\mathcal{D}$, where $\varepsilon$ is the time intervals between the
                current and previous user
12          end if
13      end if
14      if the number of tuples in $\mathcal{D} \geq \mathcal{C}$ then
15          Sample random minibatch $\mathcal{K}$ of tuples from $\mathcal{D}$
16          Calculate loss function $\mathcal{L} = \left(r_t + \gamma \max_{\boldsymbol{a_t}} Q_{tot}\left(s_t, \boldsymbol{a_t} \mid \overline{\theta}\right) - Q_{tot}\left(s_{t-\varepsilon}, \boldsymbol{a_{t-\varepsilon}} \mid \theta\right)\right)^2$
17          Perform a gradient descent step on $\mathcal{L}$ to the network parameters $\theta$, $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}$, where
            $\alpha$ is defined as the neural network learning rate
18          Update target network parameters $\overline{\theta}$ by $\overline{\theta} \leftarrow \beta\overline{\theta} + (1-\beta)\theta$, where $\beta$ represents the soft
            update rate
19          Update exploration rate $\epsilon$
20      end if
21  end for
```

Compared with traditional single-agent RL methods, the MARL framework effectively quantifies the impact of NCVs and separates two different assignment strategies for CVs and NCVs, making the purpose of each agent be learned more explicitly. However, the multi-agent setups bring new challenges. The notable one is to measure the influence of each agent's action according to the single sharing reward, known as the credit assignment problem (Minsky, 1961). Inaccurate credit assignments may lead to suboptimal strategies. Besides, most multi-agent scenarios are not guaranteed to fit the convergence theory of Q-learning used in the single-agent RL as the interactions among multiple agents continually reshape the environment and confront Markov property (Hernandez-Leal et al., 2017). Value decomposition is one of the mainstream frameworks for solving such challenges, of which the QMIX algorithm is a representative one with wide applications (Nguyen et al., 2020).

The QMIX algorithm is based on value decomposition and implicitly solves the credit assignment problem through gradient backpropagation of the mixing network. The learning ability of the deep neural network guarantees the accuracy of credit assignments. The monotonicity constraint in the mixing network is a sufficient condition for the individual-global-max principle (Son et al., 2019), which ensures that the sum of the optimal strategies of agents is the optimal strategy for the system. In detail, the optimal strategy learned by A^1 should minimize the sum of travel time of all users of CVs and NCVs. The QMIX algorithm requires agents to be trained simultaneously but allows each agent to execute separately. Therefore, it is adequate to deploy only A^1 without investing additional resources for implementation.

4.4. Simulator

In this subsection, we design a simulator that explicitly describes the dynamics in an OPA system in a partially CVs environment. As shown in Algorithm 2, each time interval includes the following steps: update the occupancy status of each parking spot, implement the online assignment strategy, and monitor the results of the assignment. In detail, the simulator records and counts the remaining occupancy time if a spot is occupied at the beginning of each time interval. Furthermore, if the remaining occupancy time of a spot goes to zero, it means that the user who parked in the spot has fulfilled his or her needs and is leaving. The spot becomes available for the following demands. Due to the lack of necessary communication with NCV users, the operator does not know the exact remaining occupancy time of all spots due to those occupied by NCV users. To formulate observable states for RL agents, we estimate the occupancy time for NCV users based on the average parking duration when users arrive in the historical data. Besides, the estimated occupancy time is reset to zero immediately when the corresponding user departs in advance, and conversely, the estimated value is constant at one if the user has not yet left but counts down to zero.

When a CV user reaches the end boundary, the operator assigns an optimal parking lot to the user based on a pre-defined assignment strategy. As for the NCV users, their destination information is unobtainable, but we can infer serval possible optimal parking lots for each destination following the assumption about their optimal choices. Then, the simulator shifts an available parking spot in each optimal parking lot from “available” to “locked,” preventing the operator from assigning the subsequent users to a parking lot that is already saturated. After crossing the end boundary, the user drives and arrives at the target parking lot. At this time, the simulator shifts the status of the spot locked in the target lot to “occupied” and renews the parking spots in other parking lots from “locked” to “available”. The simulator calculates the remaining occupancy time for the occupied spot, which is equal to the sum of the current cruising time, the parking duration, and the walking time to and from the destination. It is worth noting that once the NCV user arrives at a particular parking lot, the operator can speculate on their destination based on their optimal behavior assumption.

After reaching the predetermined simulation steps, the simulator outputs the metrics of the assignment strategy, including the parking lot assigned to each user, the corresponding cruising time and walking time, and the occupancy of parking lots at each time interval. At last, the simulator resets and prepares for the next round.

Algorithm 2: Simulator for an OPA system in a partially CVs environment

```
1   **Input**: information of the historical parking demands and spatial information of parking lots and destinations
2   Initialize parking demands, including arriving time, duration, destination, and user type (CV or NCV) of each user
3   Initialize the occupancy of parking lots
4   for $t = 0$ to $\bar{t}$ **do**
5       ***Parking status update:*** update the remaining occupancy time on each parking spot if it is occupied. If the remaining occupancy time goes to zero, the
        corresponding spot becomes “available” simultaneously. Update the estimated occupancy time for inferring the departure time of NCV users as the part
        of the state that can be observed
6       **if** a user arrives at the end boundary **then**
7           ***Assignment strategies implement***: according to the operator’s assignment strategy, assign a parking lot to the CV user or speculate on several
            possible options for the NCV user
8           ***Lock parking spot***: shift an available parking spot in each target parking lot from “available” to “locked”
9       **end if**
10      ***Assignment outcomes update***: when a user arrives at a parking spot, the status of the spot is converted from “locked” to “occupied,” and other spots
        locked for that user become “available.” The remaining occupancy time is calculated along with an estimated occupancy time
11  **end for**
12  **Output**: the results of each user’s assignment (choice) and the corresponding cruising and walking time, and the occupancy of parking lots at each time
    interval
```

5. Experiments

In this section, we present simulations to demonstrate the effectiveness of the proposed approach. We first examine a small case of the downtown campus of Tongji University (Shanghai, China) to investigate the rationale for different assignment strategies and provide some managerial insights. Then, we demonstrate the applicability of our approach by considering a larger case in the Shanghai Hongqiao Integrated Transportation Hub. The proposed QMIXm algorithm with modified $\epsilon-$ greedy strategy is compared with the following baselines:

a) FCFS: first-come-first-served.
b) SO: system optimum with complete parking information.
c) DA: dynamic assignment with demand prediction.
d) QMIX: QMIX algorithm with the original $\epsilon -$ greedy strategy.
e) DQN: deep q-network algorithm with the original $\epsilon -$ greedy strategy.
f) DQNm: deep q-network algorithm with the modified $\epsilon -$ greedy strategy.

The hyperparameters in the algorithms of the QMIX, QMIXm, DQN, and DQNm are set as same: learning rate $\alpha = 0.0005$, soft update rate $\beta = 0.99$, discounting factor $\gamma = 0.99$, memory buffer capacity $\mathscr{C} = 500$, sampling batch size $\mathscr{K} = 32$. We train the model for 5000 episodes on a personal computer with Intel i9 CPU @3.6 GHz and 64 GB RAM.

5.1. Downtown campus of Tongji University

A typical circular parking management area abstracted from Siping Campus of Tongji University (Shanghai, China) is created as illustrated in Fig. 4. This region consists of 4 parking lots, 2 user destinations, and a ring-shaped V2I sensing zone with a driving time of 10 min. The maximum occupancies are marked next to the corresponding parking lots. Dotted lines with arrows represent walking from parking lots to destinations, and the unit is minute. In addition, we set the average cruising time e_p for empty parking lot p as 0.1 min. We divide a day into five time periods with reference to morning and evening peak hours: before the morning peak (BM, 0 AM ~ 7 AM), morning peak (MP, 7 AM ~ 9 AM), off-peak (OP, 9 AM ~ 6 PM), evening peak (EP, 6 PM ~ 8 PM), and after evening peak (AE, 8 PM ~ 12 PM). We categorize users into three groups, and their parking durations follow normal distributions based on the activities: working ($c^d \sim N(480, 120)$), shopping ($c^d\tilde{N}(120, 30)$), and visiting ($c^d \sim N(20, 5)$) in the unit of one minute. We set that the parking duration of each group is stable throughout the day; in other words, the above distribution remains unchanged. However, the proportion of users from different groups varies over time; overall, the distribution of parking duration is still time-dependent. Table 1 shows the detailed proportion of the parking demand corresponding to different user destinations and time periods.

Each parking lot is heterogeneous in terms of distances to destinations and capacities, and p_3 is the best choice for the majority of users. p_3 represents the high-graded resources near the destination, which is however limited by space restrictions and provides scarce parking spots. At last, we simulate a whole day and set one unit of time interval (temporal resolution) as 1 min.

5.1.1. Model training

Fig. 5 presents the cumulative rewards of over 5000 episodes for the learning agents under various CVs penetration rates. Training methods are distinguished by different colors: blue, green, red, and orange represent QMIXm, QMIX, DQNm, and DQN, respectively. The bold smoothed curves in Fig. 5 are obtained by a moving average process to highlight learning tendencies.

For different penetration rates, the agent with a higher penetration rate is under more learning strain and has more potential. The weight of the neural network is formed arbitrarily at first, and the action created might belong to the random strategy, whose

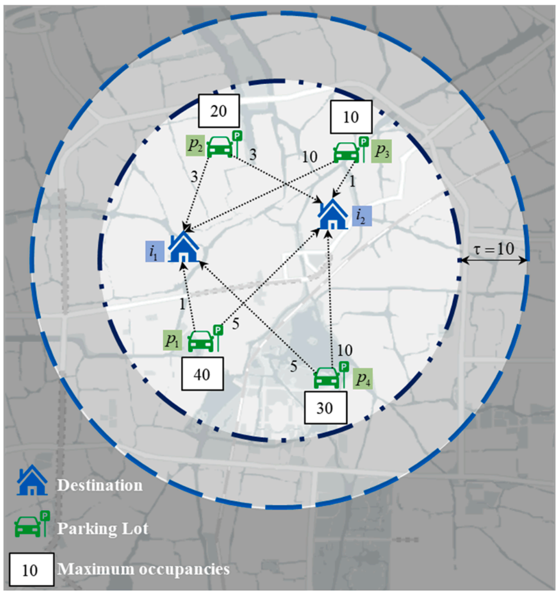


Fig. 4. Parking management area used for simulation.

Table 1
Information regarding user demand.

	BM	MP	OP	EP	AE
The quantity of parking demand					
	$N(10, 5)$	$N(30, 10)$	$N(50, 10)$	$N(30, 10)$	$N(10, 5)$
Proportion of parking demand					
Working	0.8	0.8	0.2	0.1	0.1
Shopping	0.1	0.1	0.4	0.1	0.1
Visiting	0.1	0.1	0.4	0.8	0.8
Proportion of parking demand					
i_1	0.2	0.2	0.2	0.2	0.2
i_2	0.8	0.8	0.8	0.8	0.8

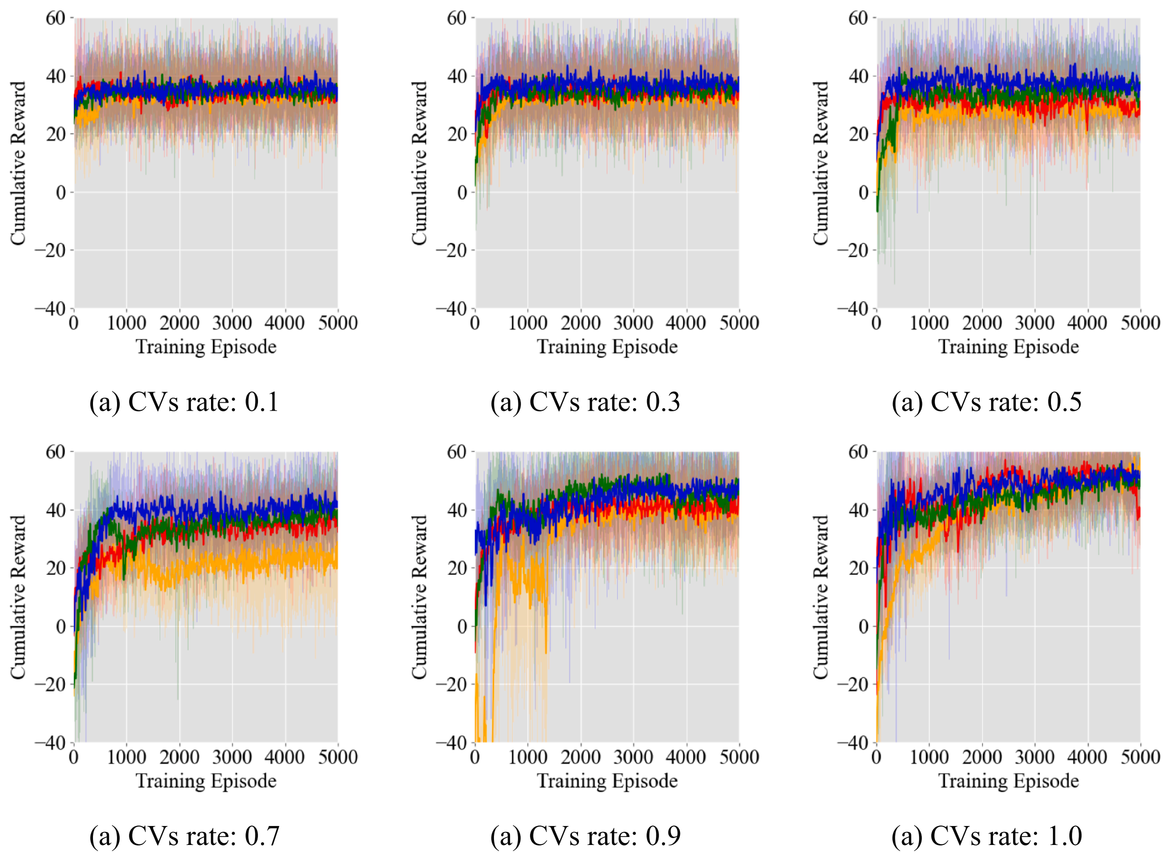


Fig. 5. Training process of the learning methods under different CVs penetration rates. Blue: QMIXm; Green: QMIX; Red: DQNm; Orange: DQN. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

performance is less practical than that under the FCFS strategy (the agent with a low penetration rate can be regarded as following the FCFS strategy). In the beginning, an agent with a higher penetration rate performs poorly on assignments. During the training process, agents gradually learn to assign CVs to appropriate parking lots, reduce total travel time, and increase accumulated rewards.

In terms of different training methods, the modified $\epsilon-$ greedy strategy allows the agent to improve without starting from a completely random assignment strategy and guides agents to learn better than the traditional exploration strategy, especially for DQNm. Compared with DQN and DQNm, QMIX and QMIXm respectively learn better and faster in scenarios of penetration rates within 0.5 and 0.9. It is likely that the QMIX structure strips the influence of NCVs from the environment, which is equivalent to alleviating the learning task of the CVs agent. In other words, the QMIX CVs agent (A^1) needs to fit the environment dynamics and learn to cooperate with the NCVs agent (A^2), while each assignment action of NCVs agent is quantifiable in the QMIX algorithm. The DQN CVs agent, on the other hand, has to consider these two types of dynamic information combined as the environment dynamics and the enormous stochasticity may hinder agent learning a good strategy or even converging. Therefore, the performance of DQN and DQNm is vulnerable to the amount of NCVs in the environment. With high CVs penetration rates or negligible influence of NCVs, the performance of QMIX, QMIXm, DQN, and DQNm are similar. In fact, QMIX is equivalent to DQN when the penetration rate is equal to 1. For the low penetration rates, the actions of the CVs agent have slight impacts due to limited assignment opportunities, and thus the differences between the four training methods are not significant.

5.1.2. *Experimental results*

For performance comparisons between QMIXm and other baselines, the following experiments are repeated 100 times separately with different random seeds to ensure that every model encounters the identical parking demand (except for CV penetration rate) in each experiment. The total travel times (of both CVs and NCVs) are presented in Fig. 6. It can be observed that the learning-based approach improves assignment performance with increasing penetration rates overall. By comparing QMIXm with QMIX, and DQNm with DQN, respectively, the results show that the modified $\epsilon-$ greedy strategy can effectively guide the agent to converge to a better assignment strategy. The QMIX structure allows the CVs agent to quantify the influence of NCVs and therefore outperforms DQN at low penetration rates (0.1, 0.3, 0.5). As the proportion of NCVs in the environment decreases, the advantage of the QMIX structure gradually diminishes, and the performance of the four algorithms is quite close with a penetration rate of 0.7 and above.

The FCFS strategy is the upper bound of OPA, and any effective OPA method should be better than it. However, it can be observed

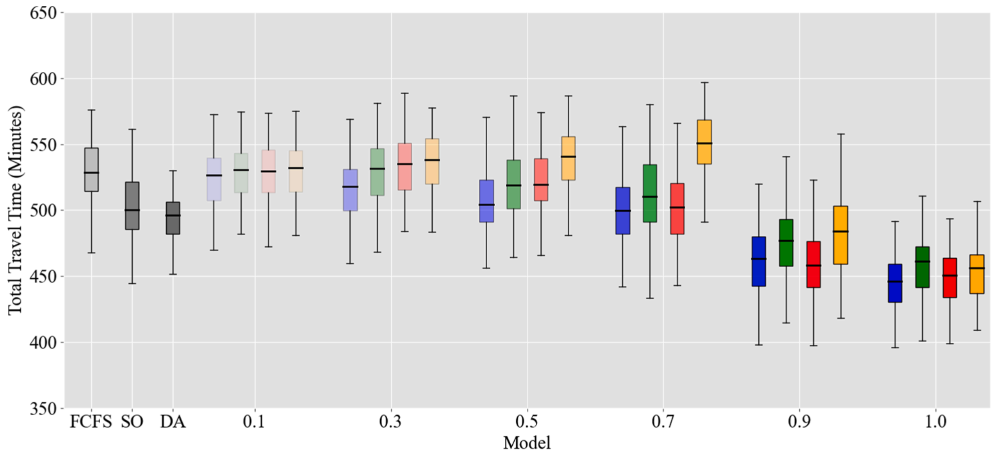


Fig. 6. Assignment performance for QMIXm and baselines in the campus. The numbers in the X-coordinates represent the penetration rates.

that only the QMIXm algorithm has a shorter average travel time than that following FCFS under various penetration rates, and the agent trained in the pure CVs environment outperform others as expected. These results demonstrate that the QMIXm algorithm, especially with a pure CVs configuration, has considerable improvements in saving users' travel time.

To further evaluate model performance, the travel time is decomposed into walking time and cruising time. The improvements on different time components are obtained in comparison with the FCFS strategy, as the results show in Table 2. The SO reduces travel time by 5.1% for all users compared to the naïve FCFS. Despite having complete user information and predictive information about the occupancy of parking lots, the SO baseline has difficulties of quantifying the dynamic cruising time well. As mentioned before, the SO avoids assigning users to parking lots that are predicted to be occupied. This avoidance leads to increases in the occupancy of other parking lots, which is not reflected in the predicted information and the required cruising time is outdated and lower than the current

Table 2
Detailed performance comparisons.

Models	Travel Time (Minutes)	Improvement (%)	Walking Time (Minutes)	Improvement (%)	Cruising Time (Minutes)	Improvement (%)
FCFS	528.81	/	266.91	/	261.90	/
SO	501.69	5.13	212.45	20.42	289.24	−10.44
DA	495.12	6.37	228.48	14.40	266.64	−1.81
CV penetration rate:0.1						
QMIXm	524.86	0.75	260.76	2.31	264.10	−0.84
QMIX	527.79	0.19	260.68	2.34	267.11	−1.99
DQNm	530.13	−0.25	262.62	1.61	267.51	−2.14
DQN	530.31	−0.28	276.21	−3.48	254.10	2.98
CV penetration rate:0.3						
QMIXm	516.16	2.39	255.23	4.38	260.93	0.37
QMIX	531.45	−0.50	273.09	−2.31	258.36	1.35
DQNm	535.61	−1.29	284.33	−6.53	251.28	4.06
DQN	535.96	−1.35	276.09	−3.44	259.86	0.78
CV penetration rate:0.5						
QMIXm	506.54	4.21	251.46	5.79	255.09	2.60
QMIX	518.87	1.88	252.18	5.52	266.70	−1.83
DQNm	521.53	1.38	264.85	0.77	256.68	1.99
DQN	528.67	0.03	288.30	−8.01	240.37	8.22
CV penetration rate:0.7						
QMIXm	498.37	5.76	245.44	8.04	252.93	3.43
QMIX	511.76	3.22	245.61	7.98	266.14	−1.62
DQNm	499.59	5.53	248.39	6.94	251.20	4.09
DQN	548.03	−3.64	287.69	−7.78	260.35	0.59
CV penetration rate:0.9						
QMIXm	461.78	12.68	236.42	11.43	225.37	13.95
QMIX	477.83	9.64	227.09	14.92	250.73	4.26
DQNm	460.52	12.91	214.77	19.54	245.76	6.16
DQN	484.78	8.33	239.13	10.41	245.66	6.20
CV penetration rate:1.0						
QMIXm	445.81	15.70	232.07	13.05	213.74	18.39
QMIX	460.15	12.98	222.85	16.51	237.31	9.39
DQNm	448.88	15.12	205.36	23.06	243.52	7.02
DQN	452.55	14.42	223.06	16.43	229.49	12.37

value. Then, the following users will be assigned to these near-saturated parking lots and experience excessive cruising time. On the other hand, the DA can update the required cruising time of parking lots before each decision, avoiding assigning users to the near-saturated parking lot. The results show that DA effectively reduces the user's travel time by 6.4% compared with the FCFS.

In comparison to the FCFS strategy, the QMIXm algorithm successfully decreases both cruising and walking time, especially in experiments with high CVs penetration rates. On average, the QMIXm algorithm reduces walking time by 11.43% and 13.05%, cruising time by 13.95% and 18.39%, and travel time by 12.68% and 15.70% when CVs penetration rate reaches 90% and 100%, respectively. In short, the QMIXm algorithm achieves significant improvement in assignment performance after receiving user parking information of CVs shortly in advance.

Apart from the macroscale inspection, we conduct a further evaluation on a microscale level, focusing on the detailed assignment decisions of the QMIXm algorithm. The dynamic occupancy of each parking lot is depicted in Fig. 7, where the curves are used to illustrate the average occupancy, and the color represents the type of user demand accepted by the parking lot on average. As observed, there are roughly three ways to assign parking lots in general: the FCFS strategy (or the QMIXm algorithm with low penetration), the QMIXm algorithm with high penetration, and the SO strategy.

The FCFS strategy is relatively straightforward. During the BM period, all demand is directed to the nearest and empty p_3. However, because of the limited supplies in p_3, occupancy climbs rapidly. Thereafter, it takes a long time to find an available parking spot during the MP period, and the most desirable parking lot is shifted to p_2, which requires a longer walking time but a much shorter cruising time and thus receives most of the parking demand. p_2 and p_3 have similar travel times in the OP period, so they receive parking demand together. Furthermore, this period mainly consists of shopping and visiting demand. Parking duration is reduced compared to that for working demand in the preceding period, implying that the turnover rate of parking spots is raised relatively, resulting in a downward tendency in occupancy. During the EP period, massive visiting demand arrives, and the total travel times associated with p_2 and p_3 are inherited from the previous period; thus, they share the majority of demand for the period as usual. In the AE period, the demand gradually disappears, and new arriving demand is assigned to p_3.

Compared with the FCFS strategy, the assignment decisions from the QMIXm algorithm, especially in the early stages, seem to be more strategic. In the BM period, all parking demand is primarily allocated to p_2. During the MP period, the agent continues to assign the majority of the working demand to p_2 and utilizes p_1 to compensate for the increased cruising time generated by high saturation in p_2. It is worth noting that p_3 is still empty at the beginning of this period and thus all shopping and visiting demand can be accommodated. p_3 still receives the bulk of demand in the OP period thanks to sufficient supplies and high turnover owing to short parking durations, while p_1 and p_2 serve as backups. During the EP period, p_3 receives almost all visiting demand, while p_2 receives the remainder if any. In the AE period, similar to the FCFS strategy, new arriving demand is assigned to p_3.

The subplot (h) of Fig. 7 indicates that the SO strategy assigns most users to the vacant parking lot p_2 during the MP period because it knows in advance that the parking lot p_3 should be highly occupied. However, it backfires. Assigning users to the parking lot p_2 makes p_3 empty and significantly increases the occupancy of p_2. Users need to spend extra cruising time in finding an available spot in p_2, which also can be seen in Table 2.

In general, the QMIXm algorithm assigns most of the shopping and working demand to p_2 and uses a part of the resources in p_1 to

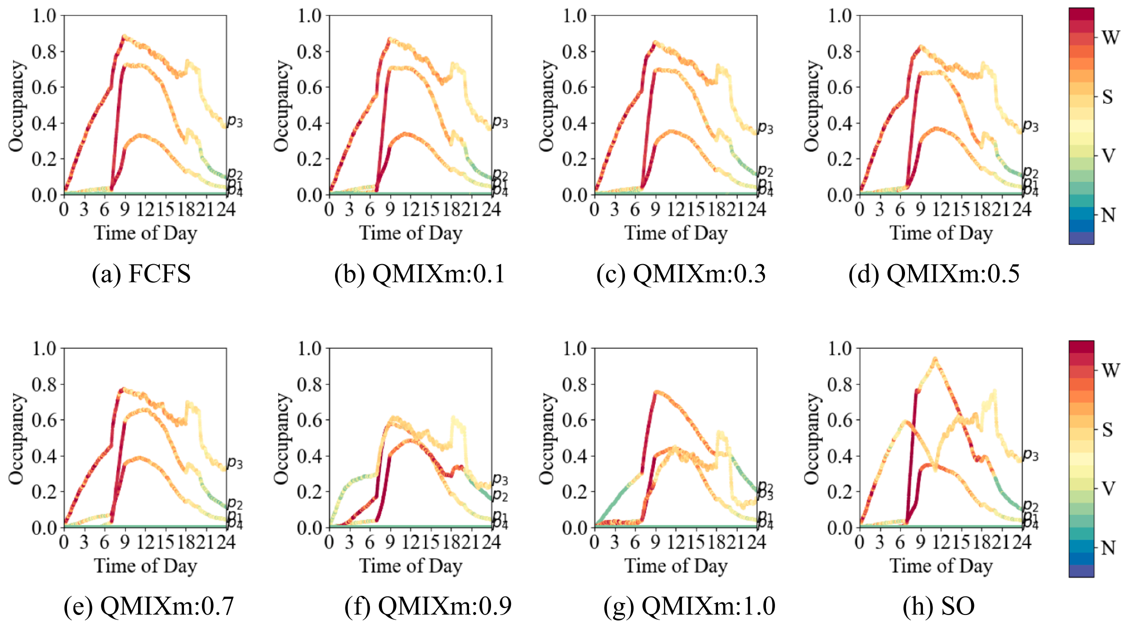


Fig. 7. Average occupancy of each parking lot. W, S, V, N in the color bar are short for working, shopping, visiting, and no user demand, respectively.

avoid the over-saturated problem in p_2. Therefore, p_3 is continually under an appropriate occupancy level so that the visiting demand can enjoy the benefits of both short walking time and cruising time.

According to the analysis above, the improvement of the QMIXm over the FCFS strategy is achieved by the mandatory assignment of shopping and working demand to further parking lots even when a better lot is available. There would be an unfair arrangement for CVs since NCVs actively seek the optimal parking lots instead of accepting allocations. Therefore, it is interesting to compare the travel time difference between CVs and NCVs, and the results are displayed in Fig. 8.

It can be seen that for all results generated by the QMIXm, the travel time of NCVs is less than that of CVs. Obviously, the working CVs take the furthest detour, and the travel time grows with an increase in the penetration rate. The return is that the travel time of shopping and visiting demand is significantly reduced. The QMIXm algorithm essentially depends on the acceptance of the assignment orders by the working CVs. In order to make the parking assignment system operate effectively, it is practical to implement supplemental policies (for example, parking fee reduction, mall coupons, etc.) to compensate for the working CVs.

Considering the possibility that not all CVs may follow the assignments, we investigate the impact of the proposed QMIXm algorithm under different obedience rates. The results are shown in Fig. 9. First of all, the scalability of different pre-trained models is not at the same level, with the QMIXm performing the worst with a 0.1 penetration rate and the best with a full penetration rate. With a 0.1 penetration rate, the QMIXm algorithm may suffer from excessively unpredictable NCVs, preventing the agent from learning. Second, both obedience and penetration rates impact the assignment performance, and these effects are similar; therefore, the contour lines form a concentric cluster with the upper right corner as the center. It is because we have configured both CVs and NCVs to be sampled from the same distribution. As a result, the penetration rate can be regarded as the visibility of the users, while the obedience rate represents the ability to assign users. For example, in the case of 100% observations of users, only 30% of users obey the assignment (but do not know which users obey), which is equivalent to 30% of observations (but do not know which users are observed) with 100% obedience. Therefore, the actual assignment should consider the joint effects of penetration and obedience rates.

5.2. *Shanghai Hongqiao Integrated Transportation Hub*

In this subsection, we further demonstrate the applicability of our method considering a real-world parking lot in the Shanghai Hongqiao Integrated Transportation Hub. The parking lot has about 2,000 parking spots and two entrances. According to the layout of parking spots, we classify all parking spots into 13 areas, as shown in Fig. 10. The dataset contained 287,562 records of valid parking transactions (i.e., parking duration was greater than 5 min), and the statistical characteristics are shown in Fig. 11. The parking lot is visited primarily by short-time parking users (less than 2 h) throughout the day, while more than half of the users belong to long-time parking (12–20 h) who concentrate on arriving at midnight. Although the traffic volume of long-time parking users is not significant, they arrive early and occupy the spots for a long time, which leads to the massive following short-time parking users not being able to access these occupied spots. It is foreseeable that an irrational assignment strategy could have long-term adverse effects, especially when these occupied spots are near destinations or high-graded. Compared to the current FCFS strategy in the parking lot, the proposed QMIXm algorithm should provide considerable room for improvement.

Without loss of generality, we assume that the parking demand obeys a uniform distribution within one hour and follows a normal distribution in terms of quantity. Then, we assume that the distribution of parking duration is stable within an hour and that the parking duration of users arriving within this hour obeys the hourly distribution. Based on the above assumptions, we can determine the duration of each parking demand by sampling from the corresponding hour of the historical data. At last, we start the simulation from a blank parking lot and set the length of one time interval as 1 min.

The following experiments are also repeated 100 times separately with different random seeds. The total travel times are presented in Fig. 12. The results show that QMIXm outperforms other baselines in reducing the total travel time of users. Compared with the FCFS, QMIXm reduces the total travel time by 5.82–16.13% when the penetration rate of CVs reaches 0.7 or more. On the other hand, compared with other learning methods, the training results of QMIXm are more stable and reliable. Meanwhile, due to the high dynamics and randomness of the environment, it is demonstrated that DQN and DQNm perform significantly worse than QMIX and

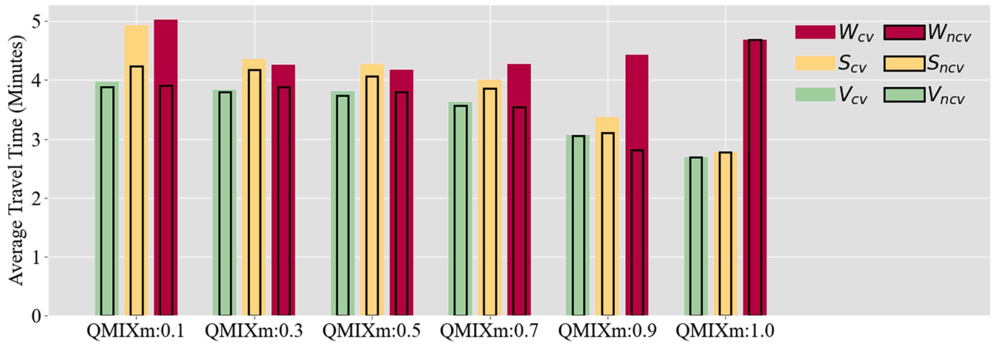


Fig. 8. The differences in travel time between CVs and NCVs. W, S, and V in the legend are short for working, shopping, and visiting demands(the subscript indicates the type of users; for example, W_{cv} represents the working CV demand).

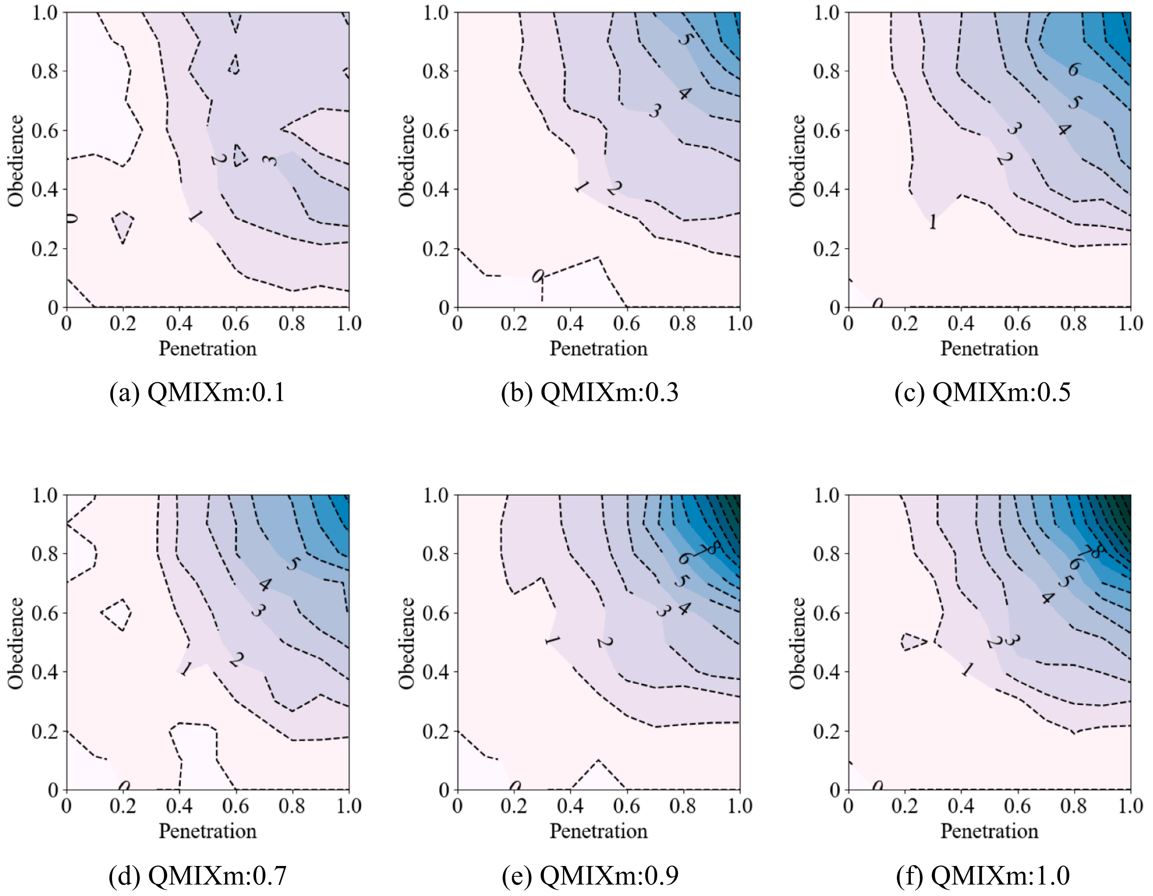


Fig. 9. Assignment performance for the QMIX under incomplete obedience. The number (%) on the contour represents the improvement compared to the results of the FCFS strategy.

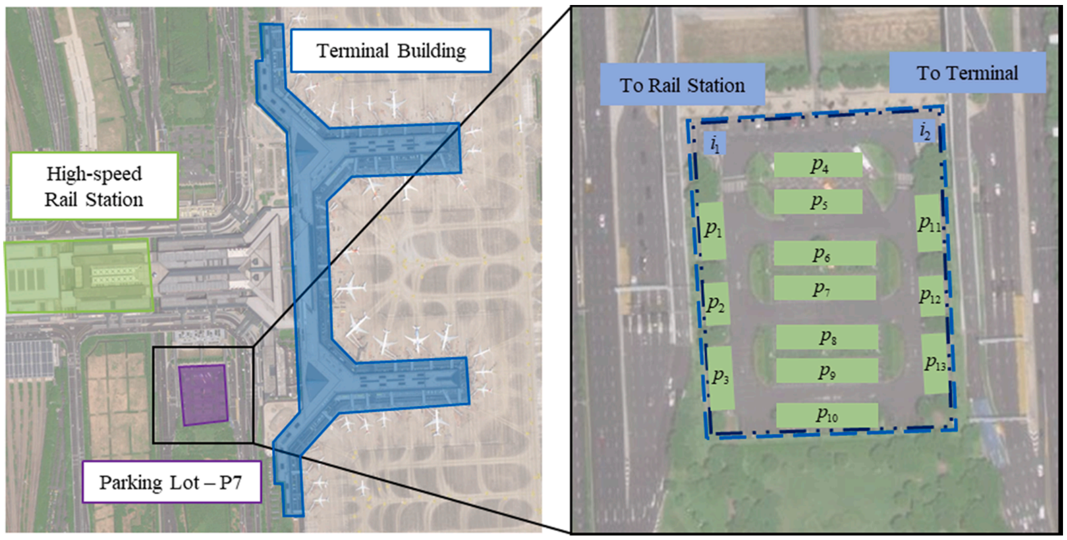
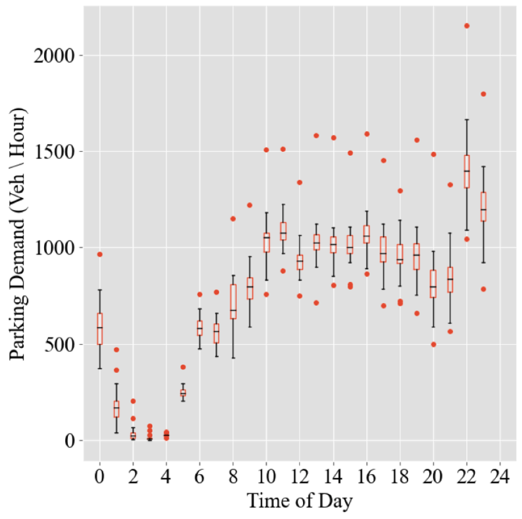


Fig. 10. The parking lot in the Shanghai Hongqiao Integrated Transportation Hub.



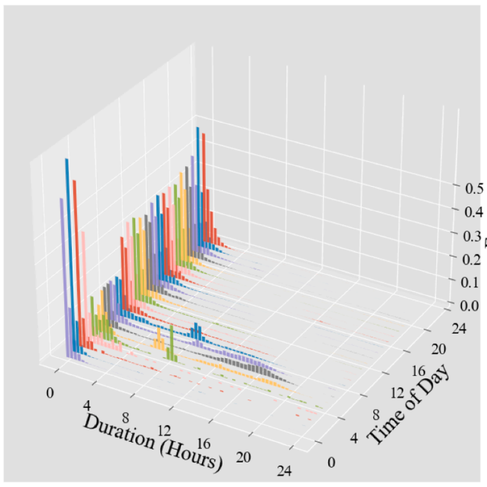


Fig. 11. The distribution characteristics of parking transaction data. (a) presents the number of users visiting the parking lot per hour and (b) indicates the distribution of parking duration within each hour.

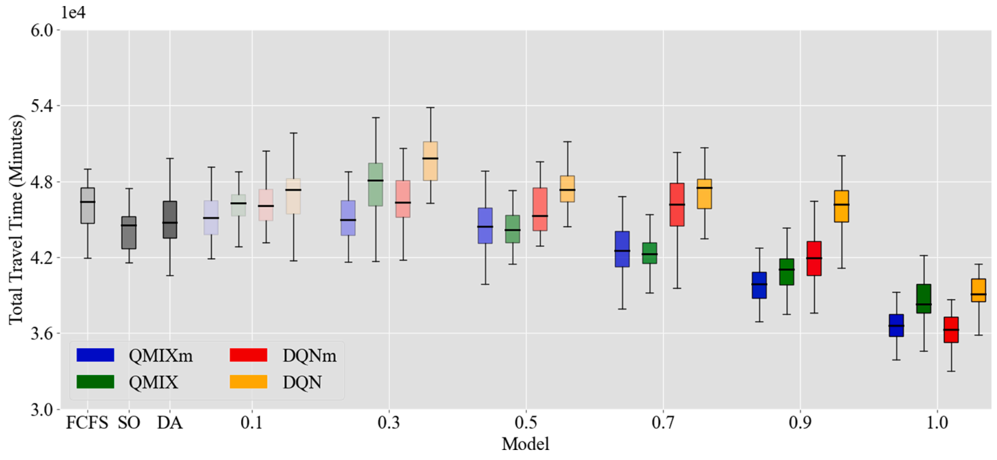


Fig. 12. Assignment performance for QMIXm and baselines in the scenario of Shanghai Hongqiao Integrated Transportation Hub. The numbers in the X-coordinate represent the penetration rates.

QMIXm in scenarios with penetration rates between 0.5 and 0.9. Finally, it is found that DA does not perform considerably better in this scenario compared to the FCFS. The main reason is that DA cannot quantify the subsequent impact of each assignment decision and easily fall into a myopic trap, while QMIXm avoids this drawback and makes each decision considering the long-term effects.

In summary, the proposed QMIXm algorithm is capable of achieving an overall high and long-term performance with partial observation information and generating dedicated assignment schemes for scenarios with different user types and parking demands.

6. Conclusion

To address the challenges of partial observations and long-term effects in the online parking assignment (OPA) problem in a mixed environment of connected vehicles (CVs) and non-connected vehicles (NCVs), this paper proposes a multi-agent reinforcement learning (MARL) framework. First, we present a fine-designed DRL formulation that includes agent appointment, state description, action selection, and a reward function. Second, to improve the efficiency of network training, we design an efficient multi-agent learning algorithm (QMIXm) with a modified exploitation strategy for directing exploration based on the nature of the problem. Third, we suggest the theoretical lower bound of the total travel time if the dynamic demands are deterministic and known beforehand. The upper bound is derived based on the FCFS strategy, which does not need any user parking information. Also, we introduce a widely used dynamic assignment (DA) method with demand prediction, which lies between the upper and lower bounds for comparisons. Fourth, we conduct a set of experiments with real applications to compare the effectiveness of the proposed QMIXm algorithm with the baselines. The results demonstrate the outperformance of the QMIXm algorithm, especially with high CV penetration rates. At last, we attempt to analyze the assignment strategy of the QMIXm and investigate the performance under different user obedience rates. The results show that the actual assignment should consider the joint effects of penetration and obedience rates.

For future work, we would first like to extend this framework to further account for parking prices, which would allow us to consider more realistic scenarios and design more comprehensive solutions. Besides, pricing management can effectively moderate users' parking behavior. Second, we assume that users have the same preferences, i.e., they have the same value of cruising time or walking time. In practice, users are generally heterogeneous. Considering the heterogeneity of users in the framework enhances the realism of implementation. Third, the number of available parking spots when the user arrives at the parking lot may be different from the number observed by the operator when assigning the users. In near-saturated cases, this may lead to detours where users cannot find available parking spots. Therefore, it is also necessary to take the driving time from the assignment point to the parking spot into account. Fourth, we sample CVs and NCVs from the same distributions, which is debatable since users with CVs and NCVs tend to have different activity-travel behavior. If the impact of CVs can be demonstrated and quantified, considering two different use distributions would be interesting. We will address these issues in our future work.

Author contribution

Xinyuan Zhang: Investigation, Methodology, Software, Writing – original draft. **Cong Zhao:** Conceptualization, Methodology, Experiments design, Software, Validation, Supervision, Writing – review & editing. **Feixiong Liao:** Formal analysis, Writing – review & editing. **Xinghua Li:** Project administration. **Yuchuan Du:** Validation, Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The study was jointly sponsored by the National Natural Science Foundation of China (52102383), the China Postdoctoral Science Foundation (2021M692428), the Innovation Program of Shanghai Municipal Education Commission (2021-01-07-00-07-E00092), the Scientific Research Program of Shanghai Municipal Science and Technology Commission (21DZ1205100), and the R&D Program of China Communication Construction Company Ltd. (2019-ZJKJ-ZDZX02). The work of Cong Zhao was supported by the Shanghai Sailing Program (21YF1449400). The authors would like to thank the anonymous reviewers for their constructive comments.

References

Ayala, D., Wolfson, O., Xu, B., Dasgupta, B., Lin, J., 2011. Parking slot assignment games. GIS Proc. ACM Int. Symp. Adv. Geogr. Inf. Syst, 299–308.
Chatman, D.G., Manville, M., 2014. Theory versus implementation in congestion-priced parking: An evaluation of SFpark, 2011–2012. Res. Transp. Econ. 44, 52–60.
Chen, I.M., Zhao, C., Chan, C.Y., 2019a. A deep reinforcement learning-based approach to intelligent powertrain control for automated vehicles. 2019 IEEE Intell. Transp. Syst. Conf. ITSC 2019, 2620–2625.
Chen, Z., Spana, S., Yin, Y., Du, Y., 2019b. An Advanced Parking Navigation System for Downtown Parking. Networks Spat. Econ. 19 (3), 953–968.
Chen, Z., Yin, Y., He, F., Lin, J.L., 2015. Parking reservation for managing downtown curbside parking. Transp. Res. Rec. 2498 (1), 12–18.
Du, L., Gong, S., 2016. Stochastic Poisson game for an online decentralized and coordinated parking mechanism. Transp. Res. Part B Methodol. 87, 44–63.
Du, Y., Chen, J., Zhao, C., Liu, C., Liao, F., Chan, C.Y., 2022. Comfortable and energy-efficient speed control of autonomous vehicles on rough pavements using deep reinforcement learning. Transp. Res. Part C Emerg. Technol. 134, 103489. https://doi.org/10.1016/j.trc.2021.103489.
Geng, Y., Cassandras, C.G., 2013. New "smart parking" system based on resource allocation and reservations. IEEE Trans. Intell. Transp. Syst. 14 (3), 1129–1139.
Hampshire, R.C., Shoup, D., 2018. What share of traffic is cruising for parking? J. Transp. Econ. Policy 52, 184–201.
He, F., Yin, Y., Chen, Z., Zhou, J., 2015. Pricing of parking games with atomic players. Transp. Res. Part B Methodol. 73, 1–12.
Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., Meger, D., 2018. Deep reinforcement learning that matters. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1.
Hernandez-Leal, P., Kaisers, M., Baarslag, T., Munoz de Cote, E., 2017. A survey of learning in multiagent environments: Dealing with non-stationarity, arXiv preprint arXiv:1707.09183, pp. 1–64.
Jiang, H., Hu, J., An, S., Wang, M., Park, B.B., 2017. Eco approaching at an isolated signalized intersection under partially connected and automated vehicles environment. Transp. Res. Part C Emerg. Technol. 79, 290–307.
Jiang, L., Molnár, T.G., Orosz, G., 2021. On the deployment of V2X roadside units for traffic prediction. Transp. Res. Part C Emerg. Technol. 129, 103238. https://doi.org/10.1016/j.trc.2021.103238.
Jiang, Z., Fan, W., Liu, W., Zhu, B., Gu, J., 2018. Reinforcement learning approach for coordinated passenger inflow control of urban rail transit in peak hours. Transp. Res. Part C Emerg. Technol. 88, 1–16.
Ke, J., Xiao, F., Yang, H., Ye, J., 2020. Learning to delay in ride-sourcing systems: a multi-agent deep reinforcement learning framework. IEEE Trans. Knowl. Data Eng. 4347, 1.
Kotb, A.O., Shen, Y.-C., Zhu, X.u., Huang, Y.i., 2016. iParker-A new smart car-parking system based on dynamic resource allocation and pricing. IEEE Trans. Intell. Transp. Syst. 17 (9), 2637–2647.
Lei, C., Ouyang, Y., 2017. Dynamic pricing and reservation for intelligent urban parking management. Transp. Res. Part C Emerg. Technol. 77, 226–244.
Levy, N., Martens, K., Benenson, I., 2013. Exploring cruising using agent-based and analytical models of parking. Transp. A Transp. Sci. 9 (9), 773–797.
Li, Z., Yu, H., Zhang, G., Dong, S., Xu, C.Z., 2021. Network-wide traffic signal control optimization using a multi-agent deep reinforcement learning. Transp. Res. Part C Emerg. Technol. 125, 103059. https://doi.org/10.1016/j.trc.2021.103059.

Lin, K., Zhao, R., Xu, Z., Zhou, J., 2018. Efficient large-scale fleet management via multi-agent deep reinforcement learning. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1774–1783.

Mao, C., Liu, Y., Shen, Z.J.(Max), 2020. Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach. Transp. Res. Part C Emerg. Technol. 115, 102626.

Minsky, M., 1961. Steps toward artificial intelligence. Proc. IRE 49 (1), 8–30.

Mladenović, M., Delot, T., Laporte, G., Wilbaut, C., 2020. The parking allocation problem for connected vehicles. J. Heuristics 26 (3), 377–399.

Mladenović, M., Delot, T., Laporte, G., Wilbaut, C., 2021. A scalable dynamic parking allocation framework. Comput. Oper. Res. 125, 105080. https://doi.org/10.1016/j.cor.2020.105080.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M., 2013. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.

Mouskos, K.C., Tavantzis, J., Bernstein, D., Sansil, A., 2000. Mathematical formulation of a deterministic parking reservation system (PRS) with fixed costs. Proc. Mediterr. Electrotech. Conf. - MELECON 2, 648–651.

Nguyen, T.T., Nguyen, N.D., Nahavandi, S., 2020. Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. IEEE Trans. Cybern. 50 (9), 3826–3839.

Qian, Z., Rajagopal, R., 2015. Optimal dynamic pricing for morning commute parking. Transp. A Transp. Sci. 11 (4), 291–316.

Qin, G., Luo, Q., Yin, Y., Sun, J., Ye, J., 2021. Optimizing matching time intervals for ride-hailing services using reinforcement learning. Transp. Res. Part C Emerg. Technol. 129, 103239. https://doi.org/10.1016/j.trc.2021.103239.

Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., Whiteson, S., 2018. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In: Proceedings of the 35th International Conference on Machine Learning. pp. 4295–4304.

Shao, C., Yang, H., Zhang, Y.i., Ke, J., 2016. A simple reservation and allocation model of shared parking lots. Transp. Res. Part C Emerg. Technol. 71, 303–312.

Shou, Z., Di, X., 2020. Reward design for driver repositioning using multi-agent reinforcement learning. Transp. Res. Part C Emerg. Technol. 119, 102738. https://doi.org/10.1016/j.trc.2020.102738.

Shoup, D.C., 2006. Cruising for parking. Transp. Policy 13 (6), 479–486.

Son, K., Kim, D., Kang, W.J., Hostallero, D.E., Yi, Y., 2019. QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In: Proceedings of the 36th International Conference on Machine Learning. PMLR, pp. 5887–5896.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W.M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J.Z., Tuyls, K., Graepel, T., 2018. Value-decomposition networks for cooperative multi-agent learning based on team reward. Proc. Int. Jt Conf. Auton. Agents Multiagent Syst. AAMAS 3, 2085–2087.

Sutton, R.S., Barto, A.G., 2018. Reinforcement learning: An introduction. MIT press.

Tang, X., Li, M., Lin, X., He, F., 2020. Online operations of automated electric taxi fleets: An advisor-student reinforcement learning framework. Transp. Res. Part C Emerg. Technol. 121, 102844. https://doi.org/10.1016/j.trc.2020.102844.

Thompson, R.G., Takada, K., Kobayakawa, S., 2001. Optimisation of parking guidance and information systems display configurations. Transp. Res. Part C Emerg. Technol. 9 (1), 69–85.

Wang, J., Sun, L., 2020. Dynamic holding control to avoid bus bunching: A multi-agent deep reinforcement learning framework. Transp. Res. Part C Emerg. Technol. 116, 102661. https://doi.org/10.1016/j.trc.2020.102661.

Wang, J., Wang, H., Zhang, X., 2020. A hybrid management scheme with parking pricing and parking permit for a many-to-one park and ride network. Transp. Res. Part C Emerg. Technol. 112, 153–179.

Watkins, C.J., Dayan, P., 1992. Q-learning. Mach. Learn. 8(3-4), 279-292.

Wang, Y., Li, M., Lin, X., He, F., 2021. Online operations strategies for automated multistory parking facilities. Transp. Res. Part E Logist. Transp. Rev. 145, 102135. https://doi.org/10.1016/j.tre.2020.102135.

Zargayouna, M., Balbo, F., Ndiaye, K., 2016. Generic model for resource allocation in transportation. Application to urban parking management. Transp. Res. Part C Emerg. Technol. 71, 538–554.

Zhang, J., Li, Z., Li, L., Li, Y., Dong, H., 2021. A bi-level cooperative operation approach for AGV based automated valet parking. Transp. Res. Part C Emerg. Technol. 128, 103140. https://doi.org/10.1016/j.trc.2021.103140.

Zhang, L., Mu, Y., 2018. Parking space allocation with uncertain demand and supply consideration. 2018 15th Int. Conf. Serv. Syst. Serv. Manag. ICSSSM 2018.

Zhao, C., Cao, J., Zhang, X., Du, Y., 2022. From search-for-parking to dispatch-for-parking in an era of connected and automated vehicles: A macroscopic approach. J. Transp. Eng. Part A Syst. 148, 1–14.

Zhao, C., Chen, I.M., Li, X., Du, Y., 2019. Urban parking system based on dynamic resource allocation in an era of connected and automated vehicles. 2019 IEEE Intell. Transp. Syst. Conf. ITSC 2019 86, 3094–3099.

Zhao, C., Li, S., Wang, W., Li, X., Du, Y., 2018. Advanced parking space management strategy design: An agent-based simulation optimization approach. Transp. Res. Rec. 2672 (8), 901–910.

Zhao, C., Liao, F., Li, X., Du, Y., 2021. Macroscopic modeling and dynamic control of on-street cruising-for-parking of autonomous vehicles in a multi-region urban road network. Transp. Res. Part C Emerg. Technol. 128, 103176. https://doi.org/10.1016/j.trc.2021.103176.

Zou, B.o., Kafle, N., Wolfson, O., Lin, J.J., 2015. A mechanism design based approach to solving parking slot assignment in the information era. Transp. Res. Part B Methodol. 81, 631–653.