Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Cross-domain coordination of resource allocation and route planning for the edge computing-enabled multi-connected vehicles

Duan Xue[1,2], Yan Guo[1*], Ning Li[1*], Xiaoxiang Song[3] and Lixiong Zhang[1]

Abstract

Multi-access edge computing (MEC) has unique interests in processing intensive and delay-critical computation tasks for vehicular application through computation offloading. However, due to the spatial inhomogeneity and dynamic mobility of connected vehicles (CVs), the edge servers (ESs) must dynamically adjust their resource allocation schemes to effectively provide computation offloading services for CVs. In this case, we propose a MEC framework supporting the collaboration of CVs, and incorporate digital twins (DTs) into wireless network to mitigate the unreliable long-distance communication between CVs and ESs. To solve the contradiction between the task change requirements of CVs and ES resources, we proactively balance the computation resources load of ESs by appropriately cooperative route planning of CVs, and achieve cross-domain load balancing between traffic flow and edge cloud resources domains. Furthermore, we jointly formulate route planning and resource allocation to balance the travel and service time delay by considering the mobility of CVs, distributed resources of ESs and the deadline sensitive vehicular tasks comprehensively. Besides, considering the coupled relationship between route planning and resource allocation, an alternating optimization algorithm is proposed to solve the formulated problem. we decompose it into two sub-problems. Firstly, a reinforcement learning method is used to optimize the route planning of CVs with fixed resource allocation. Then, an online learning and iterative algorithm is used to optimize the resource allocation strategy of edge cloud with fixed route selection. In order to demonstrate that our suggested scheme is more effective than other comparison schemes, a comprehensive series of experiments are conducted.

Keywords Multi-access edge computing, Connected vehicles, Resource allocation, Route planning, Digital twins (DTs)

Introduction

Connected vehicles (CVs) exploit on-board sensing systems and computing capabilities to understand their surroundings and drive autonomously are key to safer and more efficient transportation. CVs have difficulty processing some data-intensive tasks locally with their limited perceptual range and computational power, making it difficult to optimize traffic safety and efficiency. To address the above challenges, multi-access edge computing (MEC) has gained more attention for researchers [1–3]. MEC enables CVs to process some intensive and delay-critical computation tasks for emerging vehicular applications through offloading tasks to the edge servers (ESs) rather than executing them locally, the system performance will be enormously improved. Therefore, MEC is necessary to ensure the quality of while reducing the hardware requirements for unmanned vehicles.

*Correspondence:
Yan Guo
guoyan_1029@sina.com
Ning Li
lining_friend@sina.com
[1] College of Communications Engineering, Army Engineering University of PLA, Nanjing 210007, China
[2] College of Computer Science, Liupanshui Normal University, Liupanshui 553000, China
[3] 63891 Unit of PLA, 471000 Luoyang, China

To handle the intensive and delay-critical computation tasks of CVs offloading, some research works [4–6] suggested MEC-based architecture of task offloading for vehicular applications. These architectures deploy ESs at fixed wireless infrastructures, enabling task offloading between vehicles to ESs via vehicle-to-infrastructure (V2I) communication with low delay. In particular, the resource allocation strategy with intensive and delay-critical tasks under this architecture have attracted more attention for its practicality. In terms of reducing wireless communication delay, digital twins (DTs) [7] can realize instant wireless connectivity and extremely reliable wireless communication. DTs represent real objects or subjects in the digital space with their data, functionality and communication capabilities [8]. With the DTs, MEC can connect the physical system with the digital world in real time, enabling powerful edge intelligence in the presence of real-time changing traffic conditions. Although these references have well addressed the resource allocation problem of edge cloud, they seldom consider the impact of traffic changes on the resource allocation of edge cloud. The ES provides computation resources for CVs in a passive way, which makes it easy for vehicle terminals to fail to obtain edge resources in time, resulting in system decision-making delay and traffic accidents. This is because, on the one hand, due to the mobility of CVs, the routes change of the CVs forces the traffic flow to change in real time, making the resources required for task execution constantly change, which may make the distribution of traffic flow inconsistent with the geographical location deployment of ESs and their computational capacity distribution, affecting the performance of drive autonomously. On the other hand, due to the distribution of edge cloud, these computation-intensive tasks further exert serious pressure on some ESs with limited resources, leading to an uneven allocation of edge cloud resources, diminishing the application of MEC in intelligent transportation, and deteriorating the timeliness and correctness of offloaded tasks of CVs.

A promising solution is to proactively balance the workload of edge cloud by appropriately cooperative route planning of CVs, and jointly optimize the route planning of CVs and the resource allocation of edge cloud by real-time adjustment of edge cloud resources allocated for changes in the demand of dynamic route planning services of CVs through dynamic resource allocation techniques. It is challenging to jointly optimization of the route planning of CVs and the resource allocation of ESs. Firstly, the two variables of route planning and resource allocation are coupled with each other. For example, when the vehicular route changes, the optimal resource allocation may change accordingly. Secondly, joint optimization is actually the cross-domain resource allocation between the traffic flow and the edge cloud resource domains, which requires reasonably planning the route of CVs in the traffic flow domain and optimal resource allocation in the edge cloud resource domain. Therefore, it is necessary to balance the service delay of MEC and the travel time of CVs.

Based on the limitations of existing studies, the advantages of MEC and the communication reliability of DTs, we start from the perspective of cross-domain resource allocation, propose a joint optimization method of cross-domain resource to actively balance the resource load of edge cloud by using the route planning of CVs, and to find an effective edge cloud resource allocation while conducting reasonable route planning for CVs. In view of this, this paper focuses on the joint optimization of route planning and resource allocation to improve the quality of automatic driving of CVs in transportation system. Our work is different from previous related work in two aspects: (1) We use the collaborative route planning between CVs to actively balance the resource load of the edge cloud and achieve cross-domain load balancing between the traffic flow and the edge cloud resource domains to improve resource utilization. (2) Due to the mutual interference of CVs competing for the same edge cloud resource, we consider dynamically adjusting the edge cloud resource allocation strategy to adapt to the change of service demand of collaborative route planning of CVs, balancing the service delay of MEC and the travel time of CVs. This study makes the following contributions, which are outlined below:

- The driving routes variation of CVs and the distribution characteristics of edge cloud resources are comprehensively considered, and a joint optimization method of cross-domain resources is proposed in the MEC framework supporting the collaboration of CVs, which actively balances the resource load of edge cloud by using the collaborative route planning between CVs to achieve cross-domain load balancing between the traffic flow and the edge cloud resource domains.
- An alternating iterative optimization method is proposed to solve the coupling relationship between route planning and resource allocation. Firstly, a reinforcement learning method is used to optimize the route planning of CVs with fixed resource allocation. Then, an online learning and iterative algorithm is used to optimize the resource allocation strategy of edge cloud with fixed route selection.
- A comprehensive series of experiments are carried out to prove that our suggested scheme improves the conventional approaches in relation to average service delay, travel time and completion rate.

The structure of the paper is organized as follows: A review of relevant literatures on solving resource allocation and traffic flow optimization are presented in "Related work". Then, "System model and problem formulation" describes the edge computing model, transportation model and optimization model. In "Design and analysis of algorithm", the specific solution algorithm is given. Simulation results are illustrated in "Experimental simulation". Finally, "Conclusion and future work" discusses and summarizes the paper.

Related work

MEC for vehicle

The development of Intelligent Transportation Systems (ITS) applications has facilitated the exploration of using MEC paradigm to meet the intensive computation demand of CVs [9–11]. Luo et al. [12] proposed the use of dedicated short-range communication (DSRC) in MEC to share data between vehicles, verified that it is feasible to offload intensive and delay-critical tasks to the ESs located at 5G base stations, and done some work in improving road safety. Liu et al. [13] proposed an SDN-based heterogeneous vehicular network that provides low-delay communication in an MEC environment, thereby improving the scalability of the network. Zhang et al. [14] described the task offloading as a fatal multi-armed bandit problem, in which each round of playing with an arm is considered as choosing an ES to process the offloading task. Rodrigues et al. [15] proposed a three-tier system architecture consisting of users, multiple edge clouds and a cloud, where the edge clouds and the cloud can work together to process the user computation requests and minimize the service delay for the user. Zhang et al. [16] placed small cloud server infrastructure with limited resources such as cloudlet near the network edge and provided context-aware services based on network information. This method requires centralized frequent communication among controllers, cloudlet and mobile devices. Wang et al. [17] summarized the existing computing architecture design in MEC for CVs in detail, and analyzed the service demands and design considerations of MEC architecture from both academic and commercial viewpoints, providing a novel perspective for mobile vehicles as fog nodes, but this is just an idea. In vehicle-to-vehicle (V2V) communication, Dai et al. [18] fully utilized the computation capacity of numerous surrounding vehicles and processed tasks for closeby vehicles. However, because of the dynamic joining and departing of vehicles, the computation resources of ESs are time varying, which could cause an unforeseen delay or task failure. These works have designed MEC-based vehicular network architectures based mainly on the advantages of MEC. However, more specific offloading decision schemes and resource allocation strategies in MEC should be further investigated, which is the fundamental of vehicular networks service.

Resource allocation for MEC

Resource allocation in MEC-assisted vehicular networks has increasingly become important. However, the changes of vehicular task requirements have brought great challenges to the resource allocation of MEC. Tang et al. [19] constructed the vehicular edge computing (VEC) model in the form of frame, and proposed a dynamic framing offloading algorithm based on Double Depth Q-Network (DFO-DDQN) to minimize the total delay and waiting time of tasks from mobile vehicles. Xing et al. [20] jointly optimized the allocation and wireless resource allocation problems in MEC environment and proposed an optimal solution method based on solving a relaxed convex problem. In a FiWi-enhanced vehicular edge computing (VEC) network, Zhang et al. [21] suggested software-defined network load balancing task offloading technique. To assure a better task offloading scheme, users must bargain with one another, which necessitates constant information exchange between them. However, while minimizing the task execution delay of resource-constrained mobile users, the potential power delay trade-off is a basic challenge to be solved.

In addition, the high mobility of vehicles makes the task scheduling decisions more complicated in the presence of heterogeneous wireless and computation resources. Hence, the mobility of vehicles needs to be considered when designing resource scheduling schemes in distributed MEC scenario to support the service continuity. By using a probabilistic user mobility modeling to select the best communication channels between the base station and users and then pre-allocating the computation resources of RSUs, Plachy et al. [22] studied an excellent communication and computation resource sharing structure based on MEC. Khafajiy et al. [23] proposed a task offloading scheme with load balancing, where FSs of the Fog layer form a service group to provide services to the outside world. If an FS has a heavier load, it can also redistribute its own tasks to the lightly loaded FSs to achieve load balancing. Ouyang et al. [24] used the service migration method to increase the service quality of mobile users in the MEC environment, and proposed two heuristic algorithms based on Markov approximation and optimal response update for the solution. Considering the mobility of users, Aissioui et al. [25] proposed a resource allocation scheme that supports the migration of services across multi-edge clouds and is able to support the service continuity of vehicles.

System model and problem formulation

Scenario description

We consider an MEC empowered intelligent traffic network system as shown in Fig. 1. The traffic network has uniform RSUs deployed along the road. An ES is deployed on each roadside unit (RSU). Any one CV can only select one ES for computation offloading at the same time, but one ES can provide computation offloading services for multiple CVs at the same time. The DTs interact with physical traffic network system in real time through data collection and analytic to keep the digital plane synchronized with physical system, so that the operational state analysis and optimization of CVs can be performed directly on the ES. In order to improve the driving quality of CVs, the system not only optimizes the traffic flow but also allocates edge cloud resources.

The traffic network is defined as a directed weighted graph $\Gamma=(E,I)$ that includes a road segments set $E=\{e_1,\ldots,e_l,\ldots,e_p\}$ and $I=\{I_1,I_2,\ldots,I_q\}$ is the set of intersections. According to the lane direction, $e_i \neq e_j$, but the lengths of the road section are equal, i.e., $d_i = d_j$. A set of connected vehicles is denoted by $\mathcal{N}=\{1,2,\cdots,N\}$, and $\mathcal{M}=\{1,2,\cdots,M\}$ represents a set of ESs. Each vehicle has its own origin o to the destination d. Before the vehicle travels, ES first recommends a set of alternative routes for each vehicle based on real-time traffic conditions and shortest route algorithm. The driving routes of the vehicle i are composed of an orderly continuous road segment, which is denoted as $r_i=\{e_o,e_l,\cdots,e_d\}$. The CVs will generate computing tasks during driving, the notation $\Phi_i=\{I_i,\gamma_i,\xi_i,o_i\}$ can be used to represent the tasks of vehicle i, which includes the task input-data size I_i, the computation intensity γ_i (the number of CPU cycles required by vehicle i), the completion deadline ξ_i and the output/input ratio o_i that related to the properties of the task. To determine the computation offloading process of ESs and the dynamics of vehicles, we discretize the time into the form of non-overlapping time slots of equal length with time intervals of $\tilde{t}$. Especially, the amount of time slots is mostly determined by the maximum completion deadline, i.e., $G=\max\left\{\lfloor\xi_i/\tilde{t}\rfloor\right\}_{i\in\mathcal{N}}$. The position of vehicle i in the g-th time slot is predetermined by $p_i^g=\left(x_i^g,y_i^g\right)$. The deployment location of the ES is generally fixed, the position of the ES j is denoted as $p_j=(x_j,y_j)$.

System model

Travel route model

To indicate whether the vehicle i selects road segment $e_l \in r_i$, an indicator function is proposed as follows:

$$\beta_{i,e_l}=\begin{cases}1, & e_i=e_l\\ 0, & e_i\neq e_l\end{cases}, \tag{1}$$

where, e_i is the actual selected road segment during the traveling process of vehicle i. When the vehicle i selects the road segment e_l, β_{i,e_l}=1, otherwise, β_{i,e_l}=0.

The speed of vehicles on the road segment reflects the traffic conditions. The number of vehicles on the road

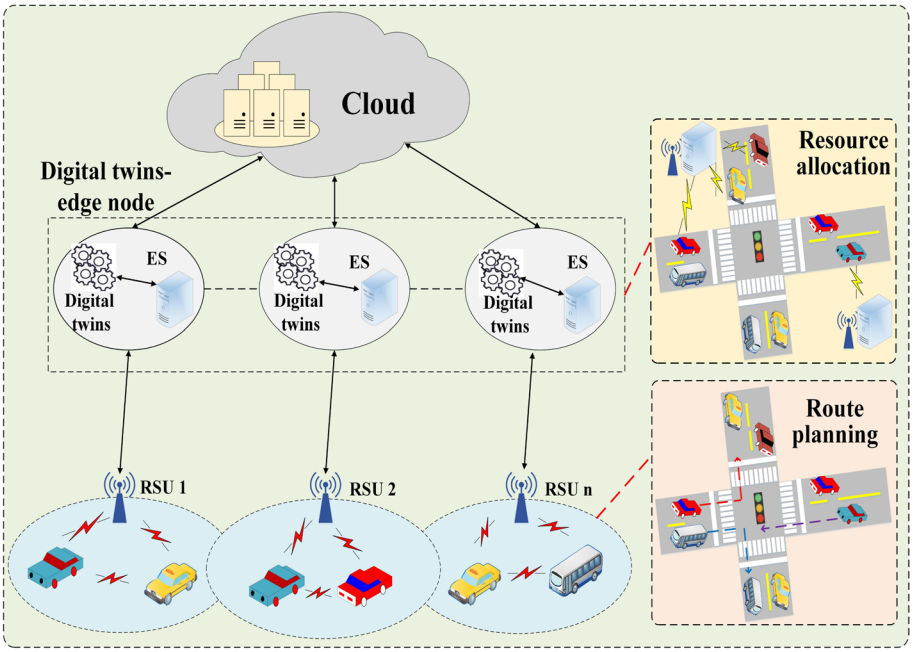

Fig. 1 CVs system framework based on MEC

segment e_l is defined as $\psi_l = \sum_{i \in \mathcal{N}} \beta_{i,e_l} + \psi_l^0$, where ψ_l^0 represents the number of vehicles backlog at the previous time slot on the road segment e_l. According to the BPR formula, the average driving speed of the road segment e_l is:

$$s_l = s_l^{free} / \left(1 + \alpha \left(\frac{\psi_l}{cap_l}\right)^{\beta}\right), \tag{2}$$

where, ψ_l represents the number of vehicles on the road segment e_l, s_l^{free} defines the free-flow speed of road segment e_l, cap_l is the capacity of the road segment e_l. α and β are adjustive parameters.

The weight ω_l of road segment e_l is calculated by the following:

$$\omega_l = \begin{cases} 1 - \frac{s_l}{\psi_l \cdot s_{l,max}} & \psi_l > 0 \\ 0 & \text{otherwise} \end{cases}, \tag{3}$$

where, $s_{l,max}$ defines the maximum speed allowed on the road segment e_l. The greater the number of vehicles, the lower the average speed, and the greater the weight of the road segment, indicating that the traffic is more congested. At this time, the weight of the road segment needs to be normalized, that is:

$$N(\omega_l) = \frac{\omega_l - \min\limits_{l \in R_v} \omega_l}{\max\limits_{l \in R_v} \omega_l - \min\limits_{l \in R_v} \omega_l}. \tag{4}$$

According to logit route selection model [26], the road segment is recommended for vehicles:

$$R(e_l) = \frac{e^{-N(\omega_l)}}{\sum\limits_{l=1}^{p} e^{-N(\omega_l)}}. \tag{5}$$

ES recommends driving routes for vehicles based on the weight of each road segment, which is different from the physical length of the road segment in previous studies and can reflect the real-time traffic variation of the road segments. When $N(\omega_i)$=0, the road segment e_l with the smallest weight will have a higher possibility of being recommended.

The recommended routes set for vehicle i is $r_i = \{e_o, e_l, \cdots, e_d\}$, where o and d represent the origin and destination, respectively. Therefore, the travel time of vehicle i on the road segment e_l is:

$$t_i^{e_l} = \frac{d_l}{s_l}. \tag{6}$$

Furtherly, the travel time of the vehicle i from o to d is:

$$t_i^{trav}(r_i) = \sum_{e_l \in r_i} t_i^{e_l}. \tag{7}$$

Communication model

Orthogonal frequency-division multiple access (OFDMA) system is used in our situation [27]. Based on the Shannon-Haley theorem, in the g-th time slot, the communication link transmission rate between ES j and vehicle i can be calculated as:

$$R_{i,j}^{g,m} = \frac{W}{\ell_j} \log_2 \left(1 + \frac{P_t^m h_{i,j}^g k_0 \left\| p_i^g - p_j \right\|^{-\theta}}{\sigma^2}\right), \tag{8}$$

where, $m = u$ for uplink while $m = d$ for downlink. the noise spectral density is denoted by σ^2, W represents the channel bandwidth, the noise spectral density is denoted by σ^2, the transmission power is represented as P_t^m, the path loss exponent is denoted by θ, and k_0 is a proportional constant coefficient that depends on $(\lambda/4\pi)^2$, the small-scale fading channel power gain between ES j and vehicle i in the g-th time slot as . The numbers of tasks running on ES j at the same time is expressed as ℓ_j:

$$\ell_j = \sum_{i \in \mathcal{N}} \alpha_{i,j}, \tag{9}$$

where, $\alpha_{i,j}$ is a binary variable, which is used to indicate whether the ES j provides computation offloading services for vehicle i, i.e., when the vehicle i selects ES j as its computation offloading object, $\alpha_{i,j} = 1$, otherwise, $\alpha_{i,j} = 0$.

Therefore, in the g-th time slot, the wireless transmission delay of the uplink for the vehicle i is:

$$t_{i,u}^{g,comm} = \frac{I_i}{R_{i,j}^{g,u}}. \tag{10}$$

Since the task has a completion deadline, the variable $D_{i,j}^{g,d}$ is introduced to represent the size of the downlink output data from ES j to the vehicle i in the g-th time slot. In the g-th time slot, the wireless transmission delay of the downlink for the vehicle i is:

$$t_{i,d}^{g,comm} = \frac{D_{i,j}^{g,d}}{R_{i,j}^{g,d}}. \tag{11}$$

Further, the wireless transmission delay of vehicle i is:

$$t_i^{comm} = \sum_{g=1}^{G} \left(t_{i,u}^{g,comm} + t_{i,d}^{g,comm}\right). \tag{12}$$

Task offloading model

As mentioned above, the computation tasks of vehicles can be offloaded to the available ESs due to limitations such as the vehicles devices. When multiple vehicles simultaneously offload computation tasks to the ES, we assume that these computation resources on the ES to be shared evenly by these tasks generated by the vehicles. When the vehicle i exceeds the coverage of the currently connected ES, the vehicle i can seamlessly switch to the ES adjacent to the ES j. As the vehicle i offloads the computation tasks to the ES j, the computation delay is that:

$$t_i^{comp} = \frac{\ell_j \cdot \gamma_i}{C_j}, \tag{13}$$

where, γ_i is the number of CPU cycles required to complete the tasks generated by the vehicle i, C_j is the computing capacity of ES j and ℓ_j represents the number of tasks simultaneously running on the ES j.

Problem formulation

The main objective of this paper is to jointly optimize resource allocation and route planning with the aim of optimizing the service delay while reducing travel time. The total time is defined as the long-term system cost, i.e., where the total cost is the service delay and travel time of the vehicle i from o to d:

$$t_i^{total} = t_i^{trav} + t_i^{comm} + t_i^{comp}. \tag{14}$$

Then, minimize the average system cost:

$$T^{ave} = \frac{1}{N} \sum_{i \in \mathcal{N}} \left(t_i^{total} \right). \tag{15}$$

Further, our problem can be expressed as the following optimization problem that:

$$\begin{aligned}
P1: \quad & \arg \min_{\substack{\alpha_{i,j}, \beta_{i,e_l}, \\ \forall i \in \mathcal{N}, \forall j \in \mathcal{M}, \forall e_l \in r_i}} T^{ave} \\
\text{s.t.} \quad & \\
& C1: \sum_{i \in \mathcal{N}} \alpha_{i,j} = 1, \forall j \in \mathcal{M} \\
& C2: \sum_{e_l \in r_i} \beta_{i,e_l} = 1, \forall i \in \mathcal{N} \\
& C3: \sum_{i \in \mathcal{N}} \alpha_{i,j} \cdot \gamma_i \le C_j, \forall j \in \mathcal{M} \\
& C4: t_i^{comm} + t_i^{comp} \le \xi_i \\
& C5: o(i), d(i) \in r_i \\
& C6: o(i) \le e_l \le d(i), \forall e_l \in r_i
\end{aligned} \tag{16}$$

Constraint $C1$ ensures that each task can only be performed by one edge server. $C2$ ensures that each vehicle can only select one road at once. $C3$ ensures that the total computation resources of the edge servers assigned to its tasks do not exceed their maximum capacity. $C4$ ensure the maximum tolerance delay for each task. $C5$ and $C6$ indicate that both o and d are on the road section, and o is ordered before the d.

Design and analysis of algorithm

It is difficult to solve the problem $P1$ directly with traditional optimization methods. On the one hand, there is an interactive coupled relationship between two optimized variables $\alpha_{i,j}$ and β_{i,e_l}, and the change of any one variable will affect the optimization of other variables. On the other hand, some key parameters needed to solve the problem are unknown, so the problem $P1$ cannot be solved directly. Therefore, to solve the coupled relationship between ES resource allocation and connected vehicles route planning, this section proposes an alternate iterative optimization method. This method first decouples the original problem $P1$ into two sub-problems. Firstly, a reinforcement learning method is used to optimize the route planning of CVs with fixed resource allocation. Then, an online learning and iterative algorithm is used to optimize the resource allocation strategy of edge cloud with fixed route selection. Finally, these two sub-problems are solved by alternate iteration, and the joint optimization of ES resource allocation and vehicle route planning is finally realized.

LRP-based route planning method

Firstly, to solve the first sub-problem, i.e., under the certain ES resource allocation strategy, adopting the LRP method to optimize the route planning, aiming at improving traffic efficiency and balancing the distribution of edge cloud requirements. The problem $P1$ can be simplified as:

$$\begin{aligned}
P2: \quad & \arg \min_{\beta_{i,e_l}, \forall i \in \mathcal{N}, \forall e_l \in r_i} T^{ave} \\
\text{s.t.} \quad & \\
& C2, C4, C5, C6
\end{aligned} \tag{17}$$

Linear reward-penalty (LRP) belongs to a matrix game distributed learning algorithm among reinforcement learning algorithms, which mainly modifies the probability distribution of each action in the action space by interacting with the environment to obtain the optimal strategy by obtaining rewards. Unlike Q-learning, which requires knowledge of the state of the environment and information about other intelligences, LRP directly uses

real-time reward information to obtain the optimal strategy in an unknown environment by iterative methods. An LRP-based route planning method to solve the problem is proposed below and the implementation process is as follows.

In this paper, the set of routes selection probability vectors for the vehicles is denoted by $\mathcal{P}_V = \{\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_N\}$. Where, $\mathbf{P}_i = \left(p_{i,e_1}, p_{i,e_2}, \cdots, p_{i,e_p}\right)$ is the routes selection probability vector for the vehicle i and p_{i,e_l} is the probability of the vehicle i selecting the road segment e_l. In each iteration, the vehicle i selects road segments according to the set of probability vectors $\mathcal{P}_V = \{\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_N\}$. After the selection is completed, for any vehicle, they respectively detect their respective total time $t_i^{total}(k)$. The reward function of the vehicle i selecting the road segment $e_l \in r_i$ in the recommended route set is denoted as $u_{i,e_l}(k)$ and its update rule is expressed as:

$$u_{i,e_l}(k) = \begin{cases} t_i^{total}(k), & e_l = e_i(k) \\ u_{i,e_l}(k-1), & e_l \neq e_i(k) \end{cases}, \tag{18}$$

where, $e_i(k)$ is actual route selection by vehicle i in iteration k. When e_l is selected, the corresponding reward function value is updated, otherwise it will remain unchanged. The probability vectors is updated as follows:

$$p_{i,e_l}(k+1) = \begin{cases} p_{i,e_l}(k) + \eta_1 \tilde{u}_i(k)\left(1 - p_{i,e_l}(k)\right) \\ -\eta_2(1 - \tilde{u}_i(k))p_{i,e_l}(k), & e_l = e_{i,\max} \\ p_{i,e_l}(k) - \eta_1 \tilde{u}_i(k) p_{i,e_l}(k) \\ +\eta_2(1 - \tilde{u}_i(k))\left(\frac{1}{l-1} - p_{i,e_l}(k)\right), & e_l \neq e_{i,\max} \end{cases}, \tag{19}$$

where, η_1 and η_2 are the learning step sizes, satisfying $0 < \eta_2 < \eta_1 < 1$. $e_{i,\max}$ is the current best road segment explored by vehicle i, denoted as:

$$e_{i,\max} = \arg\max_{e_l \in r_i} u_{i,e_l}(k). \tag{20}$$

In addition, $\tilde{u}_i(k)$ is the normalized reward of vehicle i in the $k-$th iteration, which is defined as:

$$\tilde{u}_i(k) = \frac{t_i^{total}(k)}{\max\limits_{e_l \in r_i} u_{i,e_l}(k)}. \tag{21}$$

From Eq. (18) and Eq. (19) that the route selection probability vector update only requires the total time $t_i^{total}(k)$. The LRP-based route planning algorithm is given by Algorithm 1. During each iteration, first, the set of route selection probability vector set $\mathbf{P}_i = \left(p_{i,e_1}, p_{i,e_2}, \cdots, p_{i,e_p}\right)$ of the vehicle i is initialized. Then, the vehicle i selects the route according to the probability vector $\mathbf{P}_i$. Next, the reward function $u_{i,e_l}(k)$ is updated according to the delay time $t_i^{total}(k)$ detected by the vehicle i. Finally, the set of route selection probability vector of the vehicle i is further updated. Moreover, during the iterative process, there is no need to interact any information between vehicles and vehicles, between vehicles and ESs, and between ESs and ESs as well as to understand environmental information. It can be seen that the LRP-based route planning is easier to implement and is a purely distributed approach.

Theorem 1 *For any vehicle $i \in \mathcal{N}$, given any $\varepsilon > 0$ and $\delta > 0$, there exists a $\omega_0 > 0$ and a $k_0 > 0$ such that for all $\omega_2 < \omega_0$ and $k > k_0$:*

$$Pr\left\{\left[1 - p_{i,e_{i,\max}}(k)\right] < \varepsilon\right\} > 1 - \delta, \tag{22}$$

where, $e_{i,\max}$ is the current best road segment detected by vehicle i, $p_{i,e_{i,\max}}(k)$ is the probability that vehicle i selects $e_{i,\max}$.

Proof

See Appendix A.

Initialize: For each route selection e_l of vehicle i, set the initial route selection probability $p_{i,e_l} = \frac{1}{|r_i|}$; set the initial iteration $k = 0$
if $0 < p_{i,e_l} < 1$ **then**
  for k=1 : K_1 **do**
    The vehicle i selects the route according to its own probability vector $\mathbf{P}_i$
    Observe the time $t_i^{total}(k)$ of vehicle i
    Observe reward $u_{i,e_l}(k)$ of vehicle i
    Update selection probability vectors according to the Eq. (18) and Eq. (19)
    $k = k + 1$
  end for
end if

Algorithm 1 LRP based route planning

PSO-based resource allocation method

The purpose is to maximize the use of MEC by reasonably allocating the resources of the edge cloud. Taking into account the balance and effectiveness of resource allocation, particle swarm optimization (PSO) algorithm is used to solve the resource allocation problem of ES. In this process, when the vehicle selects a certain optimal driving route, the ES resource allocation decision with the minimum delay time is found through collaborative search. The collaborative search process for vehicles can be regarded as a cyclic iterative process of "detection-interaction-learning-update", which will be described in detail below. Therefore, when the vehicles select the certain driving routes, the problem $P1$ can be simplified as:

$$P3: \arg \min_{\alpha_{i,j},\forall i\in\mathcal{N},\forall j\in\mathcal{M}_i} T^{ave} \quad \text{s.t.} \quad C1, C3, C4, C5, C6 \tag{23}$$

The problem $P3$ is also difficult to solve when the information about the position of vehicle and traffic condition are unknown, and is an NP-hard problem.

Proof Since the vehicle travels in very small time slots with little change in position, we assume that this state route selection is constant such that the computation and communication delays of the vehicle are satisfied over the duration. As a consequence, the problem P1 is transformed to delay time minimization by ES optimal resource allocation and then simplify this special case with the classic bin packing problem which is a classical NP-hard problem.

Classic bin packing problem [28]: In this problem, there are a set of bins $H = \{h_1, h_2, \cdots, h_m\}$ with a capacity of $C \in \mathbb{N}$ and a set of items $Y = \{y_1, y_2, \cdots, y_n\}$, each item has a size of $\omega_i (0 < \omega_i \leqslant C)$. The goal is to pack all items in Y into as few bins as possible so that the total size of the items in each bin must not exceed the box capacity: $\sum_{y_i \in Y(h_j)} \omega_i \leq C, \forall h_j \in X$, where $Y(h_j)$ is the set of items in $h_j \in H$. For the ES resource allocation problem, each ES is a bin with a specific capacity. Because ESs may have different hardware specifications and be able to serve different numbers of vehicles with computation and communication needs. Each vehicle with computation and communication needs is considered as an item with a certain size. Therefore, the ES resource allocation problem can be described as a variable-sized vector bin packing problem. It can be seen that the ES resource allocation problem is a special case of the classical bin packing problem and is an NP-hard problem due to the fact that the classical bin packing problem is NP-hard.

PSO

The PSO is a swarm stochastic optimization algorithm that mimics a flock of birds foraging for food. In the swarm, the state of each particle consists of three components, namely position, velocity and fitness, and the goal of all particles is to find the position with the optimal fitness value. A real vector-based approach is used to encode the positions of the particles in the PSO, then the position vector of the particle is encoded as a one-dimensional vector $\mathbf{X} = (X_1, X_2, \cdots, X_N)$ of $1 \cdot N$ with velocity denoted as $\mathbf{V} = (V_1, V_2, \cdots, V_N)$, and for any particle i, its fitness function value is denoted as $f(X_i)$, and the position with the maximum fitness value is defined as the optimal position. For particle i, the current optimal position is denoted as $X_{O,i}$. For the entire particle swarm, the global optimal position is denoted as X_G. According to the reference [29], the velocity surge is limited by adding an inertia weight in front of the velocity, and at time step k, the update equation of the PSO is given by:

$$\begin{aligned}\mathbf{V}_i^{(k+1)} = \omega \mathbf{V}_i^{(k)} &+ \kappa_1 r_1 \left(\mathbf{X}_{O,i}^{(k)} - \mathbf{X}_i^{(k)}\right) \\ &+ \kappa_2 r_2 \left(\mathbf{X}_G^{(k)} - \mathbf{X}_i^{(k)}\right)\end{aligned}, \tag{24}$$

$$\mathbf{X}_i^{(k+1)} = \mathbf{X}_i^{(k)} + \mathbf{V}_i^{(k+1)}, \tag{25}$$

where, κ_1 and κ_2 are the acceleration factors, which are non-negative real numbers. ω is the inertia weight, which takes values in the range $0 < \omega < 1$. r_1 and r_2 are random numbers between 0 and 1, and their values are updated at each iteration.

PSO-based multi-ES resource allocation

In a traffic scenario with M ESs and N vehicles, each vehicle acts as a particle to find the position with the best fitness value through collaborative search, i.e., find the ES resource allocation decision with the smallest delay time, when solving the established ES resource allocation optimization model using the PSO. The collaborative search of vehicles can be seen as a cyclic iterative process of "detection-interaction-learning-update". Among them, "detection" refers to the detection of the objective function $t_{i,\min}^{total}$ that minimizes the delay time of each vehicle in the traffic network. "Interaction" refers to the detection of the corresponding ES that minimizes the delay time allocated by the interaction between vehicles. "Learning" means getting the next ES resource allocation decision based on the update equation of the PSO. "Update" means detecting the vehicle moving to the next ES resource allocation decision through the virtual agent control system.

Firstly, to satisfy the constraint $C3$, a degree of constraint violation $viol_i$ is introduced to restrict ES to provide services for vehicles when it exceeds its task load, which is given by:

$$viol_i = \left(0, \max\left(\sum\nolimits_{i\in\mathcal{N}} \alpha_{i,j} - \left\lceil \frac{C_j}{\gamma_i} \right\rceil\right)\right), \tag{26}$$

where, C_j represents the maximum load of tasks that ES can handle at the same time.

At the same time, in order to ensure that the position vector of each particle in PSO is in the specified domain, and to ensure the effectiveness and feasibility of the algorithm, a relaxation variable $\sigma(k)$ is introduced to limit

it. As the feasible solutions increases, σ will adaptively decrease, which is:

$$\sigma(k+1) = \sigma(k)\left[1 - \frac{N_{fp}(k)}{N}\right], \tag{27}$$

where, $N_{fp}(k)$ represents the total number of temporarily feasible vehicles in the swarm after the k-th iteration, and $\sigma(0)$ is the average of the total constraint violation degree of all vehicles at the initial stage.

Then, during the detection process, the delay time detected by the vehicle i is $t_i^{(k)}$, and the corresponding ES resource allocation decision is recorded as $X_i^{(k)}$. In addition, he minimum delay time detected by the vehicle i in the k-th iteration is noted as $t_{O,i}^{(k)}$ and the corresponding ES optimal resource allocation decision is noted as $X_{O,i}^{(k)}$. The update equation is as follows:

$$t_{O,i}^{(k)} = \begin{cases} t_{O,i}^{(k-1)}, & \mathrm{t}_i^{(k)} \leqslant t_{O,i}^{(k-1)} \\ \mathrm{t}_i^{(k)}, & \mathrm{t}_i^{(k)} > t_{O,i}^{(k-1)} \end{cases}, \tag{28}$$

$$X_{O,i}^{(k)} = \begin{cases} X_{O,i}^{(k-1)}, & \mathrm{t}_i^{(k)} \leqslant t_{O,i}^{(k-1)} \\ X_i^{(k)}, & \mathrm{t}_i^{(k)} > t_{O,i}^{(k-1)} \end{cases}. \tag{29}$$

Next, when all the vehicles have completed the detection, the vehicles interact with their current minimum delay time and the corresponding ES optimal resource allocation decision. After the information interaction is completed, each vehicle can know the current global minimum time $T_G^{(k)}$ and the corresponding ES optimal resource allocation decision $X_G^{(k)}$, which are respectively represented as:

$$T_G^{(k)} = \frac{1}{N}\min\left(\sum_{i=1}^{N} t_{O,i}^{(k)}\right), \tag{30}$$

$$X_G^{(k)} = \underset{\mathbf{X}_{O,i}^{(k)}, i\in\mathcal{N}}{\arg}\ \frac{1}{N}\min\left(\sum_{i=1}^{N} t_{O,i}^{(k)}\right). \tag{31}$$

Finally, the detection vehicles are computationally updated by Eqs. (24) and (25) with their velocity and ES optimal resource allocation decision at the next iteration. Driven by this velocity update, the detection vehicle gradually converges to the global ES optimal allocation decision X_G. The PSO-based multi-ES resource allocation algorithm is described in Algorithm 2. During each iteration, first, the particle swarm is initialized, and the particle velocity and position are updated to further calculate the constraint violation degree of the vehicle. Next, the feasibility of vehicles in the swarm is determined according to Eqs. (26) and (27). Finally, the resource allocation strategy is updated.

Theorem 2 *For any detection vehicle $i \in \mathcal{N}$, if its current corresponding ES optimal resource allocation decision $\mathbf{X}_{O,i}^{(k)}$ and global ES allocation decision $\mathbf{X}_G^{(k)}$ are unchanged, then the corresponding ES allocation decision of the detection vehicle i converges to the position $\mathbf{X}_{co,i} = \frac{\left(r_1\mathbf{X}_{O,i}^{(k)} + r_2\mathbf{X}_G^{(k)}\right)}{(r_1+r_2)}$ with probability 1.*

Proof
See Appendix B.

Lemma 1 *For any vehicle $i \in \mathcal{N}$, its position sequence $\left\{\mathbf{X}_i^{(k)}\right\}_{k=0}^{\infty}$ will finally converge to $\mathbf{X}_G^{(k)}$.*

Proof
See Appendix C.

Initialize: An initial swarm is randomly generated according to the coding method given above; set $k = 0$; According to Eq. (26) calculate the relaxation variable $\sigma(0)$
for k=1 : K_2 **do**
  $N_{fp} = 0$
  for $i = 1 : N$ **do**
    Calculate $viol_i$ according to Eq. (28)
    if $viol_i \leq \sigma(k)$ **then**
      Vehicle i is regarded as a temporarily feasible vehicle
      Set $N_{fp} = N_{fp}$+1
    else
      Vehicle i is regarded as a temporarily infeasible vehicle, and N_{fp} remains unchanged
    end if
    Update the optimal position $X_{O,i}^{(k)}$ according to Eq. (29)
    Update the global optimal $X_G^{(k)}$ according to Eq. (30)
    Calculate $X_i^{(k+1)}$ according to Eq. (25)
    Update the position of all detection vehicles through the control system
    $\sigma(k) \leftarrow \sigma(k)\left[1 - \frac{N_{fp}}{N}\right]$
  end for
  $k = k + 1$
end for
Output: The global optimal position of the group is used as the ES resource allocation strategy

Algorithm 2 Multi-ES resource allocation based PSO approach

Alternating optimization algorithm

According to the Algorithms 1 and 2, an alternating optimization algorithm is designed to address the problem $P1$, i.e., joint optimization of vehicle route planning and ES resource allocation. Algorithm 3 shows the specific implementation process of the joint optimization algorithm.

1: **Initialize:** Deploy the traffic network scenario, set the number of vehicles and ESs, and assign OD pairs to each vehicle
2: **for** $K = 1 : K_{\max}$ **do**
3: Optimize the route planning under the certain resource allocation strategy by Alg. 1
4: Optimize the resource allocation under the certain driving routes by Alg. 2
5: $K = K + 1$
6: **end for**
7: **Output:** Minimum average system cost of all vehicles from origin to destination

Algorithm 3 Alternating optimization algorithm

Considering the operability and implementability in the actual traffic scenario, Fig. 2 shows the time slot division in the iterative process. The index of the time slot is noted as K and one iteration in Algorithm 3 is performed within each time slot. In which, each time slot is divided into two parts, denoted as K_1 and K_2, respectively. K_1 is used for optimizing the route planning of vehicles with Algorithm 1 and K_2 is used for optimizing the resource allocation decision of ES with Algorithm 2. To be more specific, each time slot in K_1 is further divided into four gaps. The first gap is used to recommend a driving routes for the vehicles. The second gap is used for the vehicle i to select a driving route based on the probability vector $\mathbf{P}_i = \left(p_{i,e_1}, p_{i,e_2}, \cdots, p_{i,e_p}\right)$. The third gap is used for the vehicles to detect respective minimum delay times based on the selected route information. The fourth gap is used for learning, i.e., updating the probability vectors according to Eqs. (19) and (20). Similarly, each time slot in K_2 is divided into four gaps. The first gap is used for coding swarm particles and to start detect. The second gap is used to detect the current minimum delay time of each vehicle and to update the optimal position $X_{O,i}^{(k)}$ according to Eq. (29). The third gap is used for interacting between vehicles and to update the current global optimal position $X_G^{(k)}$ according to Eq. (31). The fourth gap is used to calculate the next detection position $X_i^{(k+1)}$ for the vehicle i according to Eq. (25).

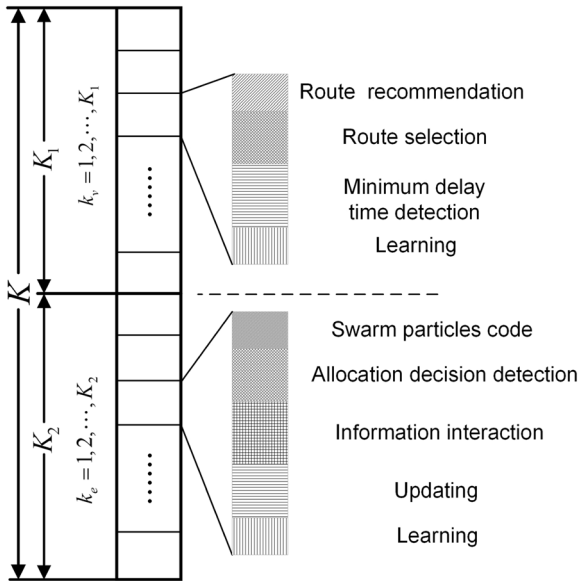

Fig. 2 Time slot model

Experimental simulation

Experiment setup

We simulate a 5km × 5km square area and assume that all the signal lamp cycles are set as static in the traffic network and the vehicles stop immediately after arriving at the destination without occupying the ES computation resources and road resources. The computation model settings and the channel model suggested in [27] are adopted in the simulation, where the small-scale fading channel power gain is in unit mean. The vehicles entering the traffic network from the origin follow the Poisson distribution process and the OD pairs are randomly assigned to each vehicle based on the normal distribution. The learning parameters are set as η_1=0.003, η_2=0.005, respectively. Table 1 provides the relevant parameters involved in the simulation. Subsequently, we implement and study the performance of the proposed alternating optimization (AO) algorithm and the comparison algorithms in MATLAB:

- Real-time route replanning (RTRR) algorithm [30]: The server calculates the k shortest routes and sends them on a first-come, first-served basis to the vehicles caught in the congestion;
- Social vehicle route selection (SVRS) algorithm [31]: The SVRS divides the vehicles into serval clusters. Subsequently, the first $n-1$ vehicles in each cluster select routs according to historical data, and the last vehicle selects the route with the shortest travel time when the other vehicle's route is provided;

Table 1 Detailed parameters of experiment

Parameter	value
W	30 MHz
σ^2	$2 * 10^{-13}$W/Hz
θ	3
P_t^m	1 W
l_i	unif(100, 1000) kbps
o_i	0.5
ξ_i	unif(500, 1500) ms
γ_i	unif(500, 1000) cycles/bit $\times$ l_i
C_j	unif(2, 6) GHz
cap_l	100 veh/km
s_l^{free}	15 m/s
$s_{l,\max}$	20 m/s
BPR α	0.15
BPR β	4

- Shortest distance first (SDF) algorithm [32]: SDF is suitable for the tasks whose importance is determined by the distance;
- Best-response allocation (BA) algorithm [24]: Formulate the resource allocation as a congestion game and use best response updates to achieve NE;
- RSU-edge system (RES) algorithm [33]: A distributed route planning service system based on MEC, in which the users' route planning tasks can be redistributed from an overloaded RSU to its neighbors RSUs.

Comparative analysis of experimental results

Algorithm performance analysis

The impact of service delay time: Vehicles need high-quality service quality during driving. Therefore, at the ES layer, communication and computation service delays need to be reduced to ensure the quality of service of the vehicles. Figure 3 shows the comparison results of average service delay time with different algorithms. From the Fig. 3, it can be seen that the AO algorithm can minimize the service delay time for traffic scenarios with different traffic flows. This is because the AO algorithm can not only plan the optimal driving routes of vehicles, but also allocate resources rationally for vehicles according to the load situation of ES. Using route planning to actively balance the resource allocation of ES without causing the resource overload of ES. Because the RES algorithm can work collaboratively between RSUs during the processing of a user's route planning task, given different neighbor levels of RSUs, reducing query response time and its performance performs better than the other two combined algorithms. However, this way of working of the RES algorithm consumes more processing time or waiting time. It can also be seen that the AO algorithm has the slowest growth rate as the traffic flow and task size increase.

The impact of travel time: Travel time is the main performance indicator to measure the driving route of vehicles in the actual traffic network. In this paper, we mainly calculate the average travel time of all vehicles to complete the driving task, as shown in Fig. 4. In the Fig. 4, the AO algorithm significantly outperforms the other two combined algorithms and the RES algorithm. The results of several algorithms perform similarly in the case of low traffic flows. This is because the traffic flows are relatively evenly distributed in the traffic network at this time and there is no potential congestion. However, as the increasing of traffic flows, the AO algorithm outperforms the other algorithms. This is because the AO algorithm can effectively direct traffic flows and distribute the vehicles to the road segments evenly. Among them, for the RES algorithm, although assigning the task of the optimal driving route to the optimal RSU can improve the resource utilization of

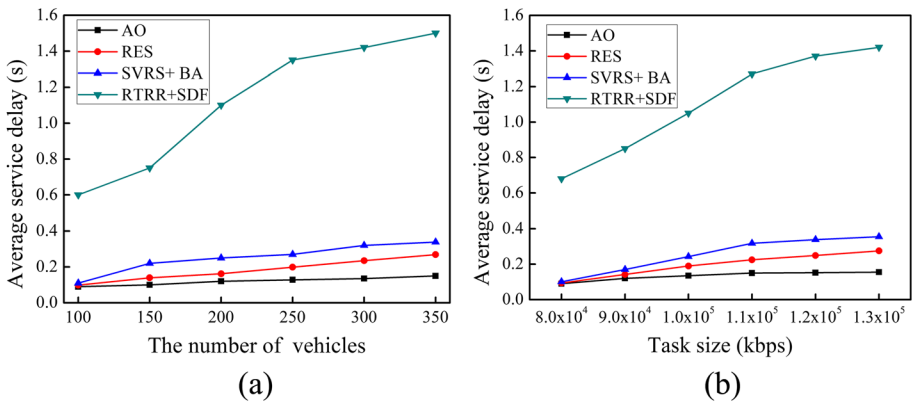

Fig. 3 Impact of service delay time. **a** w.r.t. The number of vehicles **b** w.r.t. Task size

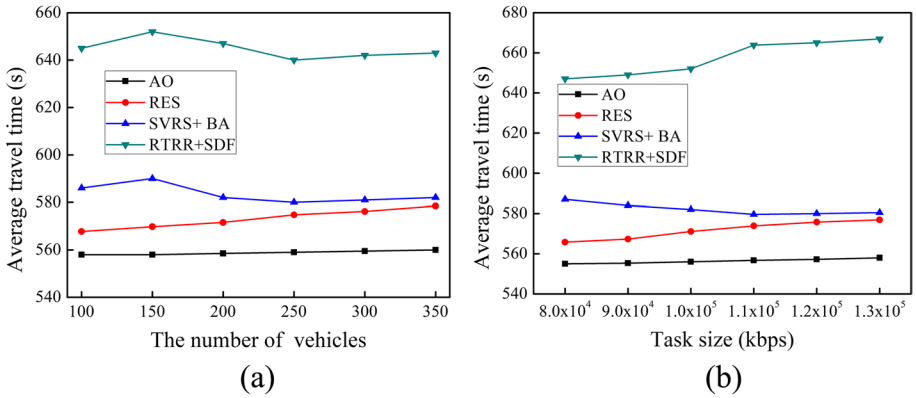

Fig. 4 Impact of travel time. **a** w.r.t. The number of vehicles **b** w.r.t. Task size

the RSU, it will lead to the resource over-utilization of some RSUs, and reassigning the task to the sub-optimal RSU will appear that the data used to generate the route is not up-to-date, which increases the possibility of road congestion and leads to an increase in travel time. Also, the AO algorithm achieves high traffic efficiency with different task size settings and can reasonably allocate effective ES to provide communication and computation services to vehicles.

The impact of completion rate under different the number of vehicles: In order to quantitatively analyze the performance of the ES resource allocation strategy, we compare the completion rate of the algorithms under different the number of vehicles. As shown in Fig. 5, the completion rate is defined as the effective return data of ES divided by the total amount of demand output data of the vehicular tasks, e.g., $CR = \sum_{i=1}^{N} \sum_{g=1}^{G} D_{i,d}^{g} \Big/ \sum_{i=1}^{N} I_i \cdot o_i$, where, $D_{i,d}^{g}$ represents the actual received downlink output data of the vehicle i in the g-th time slot. In particular, to test the advantage of the proposed AO algorithm, we adjust the distribution of task size and completion deadline in the sub-figures in Fig. 5, which are (a) $I_i = \text{unif}(100,1000)\text{kbps}, \quad \xi_i = \text{unif}(500,1000)\text{ms},$ (b) $I_i = \text{unif}(100,500)\text{kbps}, \quad \xi_i = \text{unif}(500,1000)\text{ms},$ (c) $I_i = \text{unif}(100,1000)\text{kbps}, \xi_i = \text{unif}(500,1500)\text{ms}$, respectively. In the same set of experiments, the higher the number of vehicles in the traffic network leads to higher traffic flows, but ES cannot complete more tasks than the case with fewer vehicles in the traffic network due to the limited computation resources and the same task completion deadline time. Even so, the AO algorithm still outperforms the other two combined algorithms and the RES algorithm in different traffic scenarios. The performance of the RES algorithm in terms of completion rate is similar to that the AO algorithm because the RES algorithm divides the traffic network into grids, controls the search area of RSUs based on a region of interest heuristic task allocation method, handles tasks that cannot be allocated to the optimal RSU due to overuse at the expense of vehicle travel time, and equalizes the resource utilization between RSUs. From the results, we can conclude the importance of the completion deadline time on the performance of the algorithm. It can also be seen that the difference between the

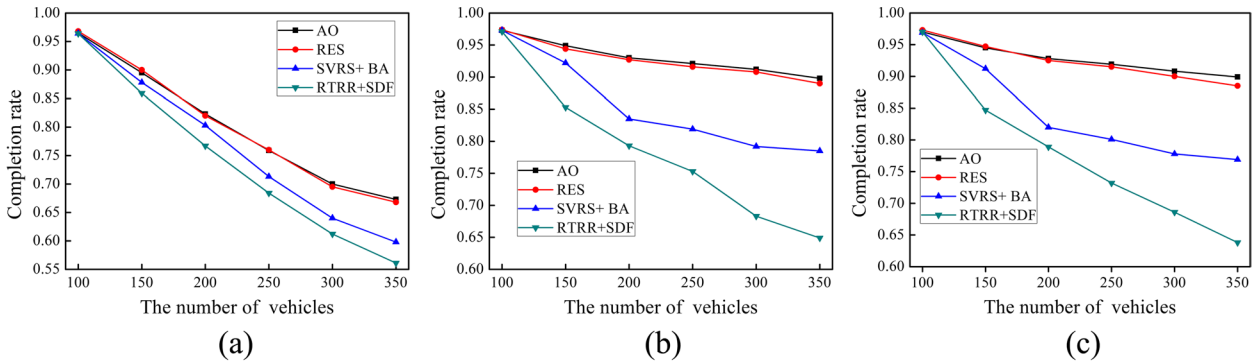

Fig. 5 Impact of completion rate. **a** $I_i = \text{unif}(100,1000)\text{kbps}, \xi_i = \text{unif}(500,1000)\text{ms}$ **b** $I_i = \text{unif}(100,500)\text{kbps}, \xi_i = \text{unif}(500,1000)\text{ms}$ **c** $I_i = \text{unif}(100,1000)\text{kbps}, \xi_i = \text{unif}(500,1500)\text{ms}$

algorithms decreases with increasing traffic flows. It is conceivable that the lines in Fig. 5 will overlap when the computation capacity of ES is sufficiently large. However, for real traffic scenarios, the AO algorithm is able to provide stable service delay and computation offloading services.

Impact of learning parameters on algorithm 1

This subsection mainly analyzes the impact of different learning parameters on Algorithm 1. Learning parameters η_1 and η_2 are important parameters that affect the convergence of the algorithm. The change of total time with iterations under different η_1 and η_2 settings is shown in Fig. 6. It can be seen that when $\eta_1 = 0.03, \eta_2 = 0.015$, the algorithm has the fastest convergence speed, but the average system cost is the longest. As the values of η_1 and η_2 decrease, the time obtained after convergence of the algorithm decreases, but its convergence speed then decreases. Therefore, a compromise between convergence speed and convergence accuracy should be considered when setting the values of η_1 and η_2.

Conclusion and future work

In this paper, we take into account the mobility of vehicles and the distribution characteristics of edge cloud resources comprehensively, and jointly optimize the problem that route planning of vehicles and the resource allocation of edge clouds under the MEC framework supporting the collaboration of CVs. An alternating iterative optimization method is designed to properly decouple the joint optimization problem, which mainly actively balances the resource load of edge cloud by using the collaborative route planning between CVs to realize cross-domain load balancing between the traffic flow and the edge cloud resource domains, and further balances the service delay of edge cloud and the travel time of CVs. First, a reinforcement learning method is used to optimize the route planning of CVs with fixed resource allocation. Then, an online learning and iterative method is used to optimize the resource allocation strategy of edge cloud with fixed route selection. Finally, a comprehensive series of experiments are carried out to prove that our suggested scheme improves the conventional approaches in relation to average service delay, travel time and completion rate. The future research direction of air-ground integrated MEC framework is also considered to take advantage of the high flexibility and mobility computing resource allocation of vehicles and UAVs.

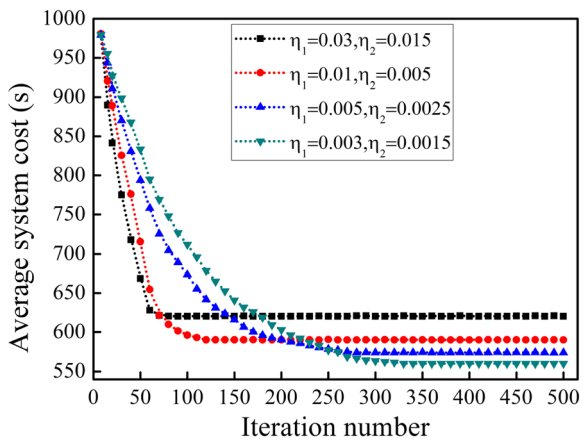

Fig. 6 Impact of learning parameters on Algorithm 1

Appendix

Appendix A

Denote Δu as the difference between the two largest expected rewards of vehicle i. The best expected reward of any vehicle i is unique and denoted as $\bar{u}_{i,e_{i,\max}}$. We have $\bar{u}_{i,e_{i,\max}} - \Delta u > \bar{u}_{i,e_l}, \forall e_l \in \{r_i \backslash e_{i,\max}\}$. Denote $\hat{u}_{i,e_l}$ as the estimate of the rewards of selecting road segment e_l obtained by the vehicle i at instant k, it follows from the weak large numbers theorem that for any given $\delta > 0$, there exists an $N_{e_l} < \infty$, such that if road segment e_l is selected at least N_{e_l} times, we have:

$$\Pr\left\{\left|\hat{u}_{i,e_l}(k) - \bar{u}_{i,e_l}\right| < \frac{\Delta u}{2}\right\} > 1 - \delta. \tag{32}$$

Let $N_{\max} = \max_{e_l \in r_i}\{N_{e_l}\}$ and denote the actual selection number of e_l until k as $Y_{e_l}(k)$, for all $e_l \neq e_{i,\max}$ and for all k, if $\min_{e_l \in r_i}\{Y_{e_l}(k)\} > N_{\max}$, then:

$$\Pr\left\{\left|\hat{u}_{i,e_{i,\max}}(k) - \bar{u}_{i,e_l}\right| < \frac{\Delta u}{2}\right\} < 1 - \delta. \tag{33}$$

Thus, if all road segments are selected at least $N_{\max}$ times, we have $\hat{u}_{i,e_{i,\max}}(k) \geqslant \bar{u}_{i,e_{i,\max}} - \frac{\Delta u}{2} > \bar{u}_{i,e_l} - \frac{\Delta u}{2}, \forall e_l \neq e_{i,\max}$, and then $\hat{u}_{i,e_{i,\max}}(k) > \hat{u}_{i,e_l}(k), \forall e_l \neq e_{i,\max}$.

According to [34], we can find a ω_0 and k_0 such that for all $\forall \omega_2 < \omega_0$ and $\forall k > k_0$:

$$\Pr\left\{\min_{e_l \in r_i}\{Y_{e_l}(k)\} > N_{\max}\right\} > 1 - \delta. \tag{34}$$

Define A and B as:

$$\begin{cases} A \equiv 1 - p_{i,e_{i,\max}}(k) < \varepsilon \\ B \equiv \max\limits_{e_l \in r_i}\left|\hat{u}_{i,e_l}(k) - \bar{u}_{i,e_l}\right| < \frac{\Delta u}{2} \end{cases}. \tag{35}$$

By the law of total probability and probability is a continuous function [34], then:

$$\lim_{k \to \infty} \Pr\{A|B\} = 1, \tag{36}$$

$$\lim_{k\to\infty} \Pr\{B\} = 1-\delta, \tag{37}$$

$$\lim_{k\to\infty} \Pr\{A\} \geqslant \lim_{k\to\infty} \Pr\{A|B\} \cdot \lim_{k\to\infty} \Pr\{B\}. \tag{38}$$

Hence, we have:

$$\lim_{k\to\infty} \Pr\left\{1 - p_{i,e_{i,\max}}(t) < \varepsilon\right\} \geqslant 1-\delta. \tag{39}$$

Appendix B

The updated Eq. (24) and Eq. (25) of the vehicle can be expressed in closed form as [35]:

$$\mathbf{X}_i^{(k)} = \mathbf{K}_1 + \mathbf{K}_2\phi_1^k + \mathbf{K}_3\phi_2^k, \tag{40}$$

where,

$$\mathbf{K}_1 = \frac{r_1\mathbf{X}_{O,i}^{(k)} + r_2\mathbf{X}_G^{(k)}}{r_1+r_2}, \tag{41}$$

$$\mathbf{K}_2 = \frac{\phi_2\left(\mathbf{X}_i^{(0)} - \mathbf{X}_i^{(1)}\right) - \mathbf{X}_i^{(1)} + \mathbf{X}_i^{(2)}}{(\phi_1-1)\sqrt{(1+\omega-r_1-r_2)^2-4\omega}}, \tag{42}$$

$$\mathbf{K}_3 = \frac{\phi_1\left(\mathbf{X}_i^{(1)} - \mathbf{X}_i^{(0)}\right) + \mathbf{X}_i^{(1)} - \mathbf{X}_i^{(2)}}{(\phi_2-1)\sqrt{(1+\omega-r_1-r_2)^2-4\omega}}, \tag{43}$$

$$\phi_1 = \frac{1+\omega-r_1-r_2+\sqrt{(1+\omega-r_1-r_2)^2-4\omega}}{2}, \tag{44}$$

$$\phi_2 = \frac{1+\omega-r_1-r_2-\sqrt{(1+\omega-r_1-r_2)^2-4\omega}}{2}. \tag{45}$$

Obviously, if $\max(\|\phi_1\|, \|\phi_2\|) < 1$, then $\lim_{k\to\infty}\phi_1^k = 0$, $\lim_{k\to\infty}\phi_2^k = 0$, so:

$$\lim_{k\to\infty} \mathbf{X}_i^{(k)} = \mathbf{K}_1 + \mathbf{K}_2\cdot 0 + \mathbf{K}_3\cdot 0 = \mathbf{K}_1. \tag{46}$$

That is, the position sequence $\left\{\mathbf{X}_i^{(k)}\right\}_{k=0}^{\infty}$ of vehicle i finally converges to $\mathbf{K}_1$. Therefore, the condition for vehicle i to converge to $\mathbf{K}_1$ is $\max(\|\phi_1\|, \|\phi_2\|) < 1$, i.e.,

$$\left\| \frac{1+\omega-r_1-r_2 \pm \sqrt{(1+\omega-r_1-r_2)^2-4\omega}}{2} \right\| < 1. \tag{47}$$

Solving the Eq. (47), we can get:

$$\frac{r_1+r_2}{2} - 1 < \omega < 1. \tag{48}$$

And for the reason of $r_1, r_2 \in (0,1)$, the upper bound of $\frac{r_1+r_2}{2} - 1$ is 0, and the value range of the inertia weight is $0 < \omega < 1$. Therefore, for any $\omega \in (0,1)$, the Eq. (48) is always true, and the position of vehicle i converges to the position $\mathbf{X}_{co,i} = \mathbf{K}_1 = \frac{\left(r_1\mathbf{X}_{O,i}^{(k)} + r_2\mathbf{X}_G^{(k)}\right)}{(r_1+r_2)}$ with probability 1.

Appendix C

According to **Theorem 2**, the vehicle i position sequence $\left\{\mathbf{X}_i^{(k)}\right\}_{k=0}^{\infty}$ will converge to $\mathbf{X}_{co,i} = \frac{\left(r_1\mathbf{X}_{O,i}^{(k)} + r_2\mathbf{X}_G^{(k)}\right)}{(r_1+r_2)}$, which can also be rewritten as $\mathbf{X}_{co,i} = (1-a_r)\mathbf{X}_{O,i}^{(k)} + a_r\mathbf{X}_G^{(k)}$, where $a_r = \frac{r_2}{r_1+r_2}$, , which takes values in the range $0 < a_r < 1$. That is, the convergence position of the vehicle i is always between $\mathbf{X}_{O,i}^{(k)}$ and $\mathbf{X}_G^{(k)}$. Also according to the update Eq. (29) of $\mathbf{X}_{O,i}^{(k)}$,it is known that $\mathbf{X}_{O,i}^{(k)}$ will gradually converge to $\mathbf{X}_G^{(k)}$. Therefore, the position sequence $\left\{\mathbf{X}_i^{(k)}\right\}_{k=0}^{\infty}$ of any vehicle i can finally converge to $\mathbf{X}_G^{(k)}$.

Acknowledgements

The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.

Authors' contributions

In this work, Duan Xue conceived and designed the communication, computation, transportation model and concrete algorithms; the idea is proposed by Yan Guo and Ning Li, and they critically reviewed the paper and contributed to the improvement on paper writing; Xiaoxiang Song critically reviewed the method used and contributed to structuring the paper; the experiments are performed by Lixiong Zhang. The author(s) read and approved the final manuscript.

Authors' information

Duan Xue received the B.S. degree from Jinggangshan University and the M.S. degree from Yunnan University University. He is currently pursuing the Ph.D. degree with the Army Engineering University of PLA, Nanjing, China. His research interests include edge computing, Internet of vehicles.

Yan Guo received the B.S. and M.S. degrees from the PLA Institute of Information Engineering, Zhengzhou, China, in 1993 and 1996, respectively, and the Ph.D. degree from Xidian University, Xi'an, China, in 2002. She has been a Visiting Scholar with Chonbuk National University, South Korea. She is currently a Professor with the Institute of Communication and Engineering, Army Engineering University of PLA, Nanjing China. Her main research interests include compressive sensing, edge computing, and cognitive radio.

Ning Li received the B.S. and M.S. degrees from the PLA Institute of Information Engineering, Zhengzhou, China, in 1989 and 1996, respectively. He is currently an Associate Professor with the Institute of Communication and Engineering, Army Engineering University of PLA, Nanjing, China. His main research interests include ad hoc networks, cloud computing, and cognitive radio.

Xiaoxiang Song received the B.S.degree from Guangxi University, China, in 2016, and the M.S. and Ph.D. degrees from the Army Engineering University of PLA, China, in 2018 and 2021, respectively. His research interests include big data, edge computing and compressive sensing.

Lixiong Zhang received the B.S.degree from the School of electrical and Electronic Engineering, Shijiazhuang Tiedao University. He is currently pursuing the M.S. degree whit the Army Engineering University of PLA, Nanjing, China. His research interests include information perception, edge computing and dynamic path planning.

Funding
This research was supported by the Natural Science Foundation of Jiangsu Province (BK20211227), National Natural Science Foundation of China (61871400, 62273356, 62261034), and Scientific Research Project of Liupanshui Normal University (LPSSYYBZK202207).

Availability of data and materials
Not applicable.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 15 August 2021 Accepted: 27 February 2023
Published online: 08 March 2023

References

1. Ali Z, Khaf S, Abba ZH et al (2021) A comprehensive utility function for resource allocation in mobile edge computing. Comput Mater Continua 66(2):1461–1477
2. Mao Y, You C, Zhang J et al (2017) A survey on mobile edge computing: The communication perspective. IEEE Commun Surv Tutor 19(4):2322–2358
3. Mach P, Becvar Z (2017) Mobile edge computing: A survey on architecture and computation offloading. IEEE Commun Surv Tutor 19(3):1628–1656
4. Ning Z, Huang J, Wang X (2019) Vehicular fog computing: Enabling real-time traffic management for smart cities. IEEE Wirel Commun 26(1):87–93
5. Cha N, Wu C, Yoshinaga T et al (2021) Virtual edge: Exploring computation offloading in collaborative vehicular edge computing. IEEE Access 9:37739–37751
6. Wu Q, Zeng Y, Zhang R (2018) Joint trajectory and communication design for multi-UAV enabled wireless networks. IEEE Trans Wirel Commun 17(3):2109–2121
7. Tao F, Zhang H, Liu A, Nee AY (2018) Digital twin in industry: State-of-the-art. IEEE Trans Ind Inf 15(4):2405–2415
8. Schluse M, Priggemeyer M, Atorf L, Rossmann J (2018) Experimentable digital twins-Streamlining simulation-based systems engineering for industry 4.0. IEEE Trans Ind Inf 14(4):1722–1731
9. Zhan W, Luo C, Wang J et al (2020) Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing. IEEE Internet Things J 7(6):5449–5465
10. Zhao L, Yang K, Tan Z et al (2021) A novel cost optimization strategy for SDN-enabled UAV-assisted vehicular computation offloading. IEEE Trans Intell Transp Syst 22(6):3664–3674
11. Liao H, Zhou Z, Kong W et al (2021) Learning-based intent-aware task offloading for air-ground integrated vehicular edge computing. IEEE Trans Intell Transp Syst 22(8):5127–5139
12. Luo G, Zhou H, Cheng N et al (2021) Software-defined cooperative data sharing in edge computing assisted 5G-VANET. IEEE Trans Mob Comput 20(3):1212–1229
13. Liu J, Wan J, Zeng B et al (2017) A scalable and quick-response software defined vehicular network assisted by mobile edge computing. IEEE Commun Mag 55(7):94–100
14. Zhang R, Cheng P, Chen Z et al (2020) Online learning enabled task offloading for vehicular edge computing. IEEE Wirel Commun Lett 9(7):928–932
15. Rodrigues TK, Liu J, Kato N (2022) Offloading decision for mobile multi-access edge computing in a multi-tiered 6G network. IEEE Trans Emerg Top Comput 10(3):1414–1427
16. Zhang F, Wang MM (2021) Stochastic congestion game for load balancing in mobile-edge computing. IEEE Internet Things J 8(2):778–790
17. Wang H, Liu T, Kim BG et al (2020) Architectural design alternatives based on cloud/edge/fog computing for connected vehicles. IEEE Commun Surv Tutor 22(4):2349–2377
18. Dai P, Hang Z, Liu K et al (2020) Multi-armed bandit learning for computation-intensive services in MEC-empowered vehicular networks. IEEE Trans Veh Technol 69(7):7821–7834
19. Tang H, Wu H, Qu G et al (2022) Double deep Q-Network based dynamic framing offloading in vehicular edge computing. IEEE Trans Netw Sci Eng. https://doi.org/10.1109/TNSE.2022.3172794
20. Xing H, Liu L, Xu J, Nallanathan A (2019) Joint task assignment and resource allocation for D2D-enabled mobile-edge computing. IEEE Trans Commun 67(6):4193–4207
21. Zhang X, Zhang J, Liu Z et al (2020) MDP-based task offloading for vehicular edge computing under certain and uncertain transition probabilities. IEEE Trans Veh Technol 69(3):3296–3309
22. Plachy J, Becvar Z, Strinati EC et al (2021) Dynamic allocation of computing and communication resources in multi-access edge computing for mobile users. IEEE Trans Netw Serv Manag 18(2):2089–2106
23. Al-Khafajiy M, Baker T, Asim M et al (2020) COMITMENT: A fog computing trust management approach. J Parallel Distrib Comput 137:1–16
24. Ouyang T, Zhou Z, Chen X (2018) Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing. IEEE J Sel Areas Commun 36(10):2333–2345
25. Aissioui A, Ksentini A, Gueroui AM, Taleb T (2018) On enabling 5G automotive systems using follow me edge-cloud concept. IEEE Trans Veh Technol 67(6):5302–5316
26. Cao A, Fu B, He Z (2019) ETCS: An efficient traffic congestion scheduling scheme combined with edge computing. In: 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). Zhangjiajie, China: IEEE; pp 2694–2699
27. Mao Y, Zhang J, Song SH, Letaief KB (2017) Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems. IEEE Trans Wirel Commun 16(9):5994–6009
28. Pisinger D (2005) Where are the hard knapsack problems? Comput Oper Res 32(9):2271–2284
29. Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6(1):58–73
30. Ahmad A, Din S, Paul A, Jeon G, Aloqaily M, Ahmad M (2019) Real-time route planning and data dissemination for urban scenarios using the internet of things. IEEE Wirel Commun 26(6):50–55
31. Lin K, Li C, Fortino G, Rodrigues JJ (2018) Vehicle route selection based on game evolution in social internet of vehicles. IEEE Internet Things J 5(4):2423–2430
32. Liu Y, Li Y, Niu Y, Jin D (2019) Joint optimization of path planning and resource allocation in mobile edge computing. IEEE Trans Mob Comput 19(9):2129–2144
33. Talusan JPV, Wilbur M, Dubey A et al (2020) Route planning through distributed computing by road side units. IEEE Access 8:176134–176148
34. Chen Z, Lin T, Wu C (2015) Decentralized learning-based relay assignment for cooperative communications. IEEE Trans Veh Technol 65(2):813–826
35. Van Den Bergh F (2007) An analysis of particle swarm optimizers. Doctoral dissertation, University of Pretoria, Pretoria

Publisher's Note