An Efficient Caching and Offloading Resource Allocation Strategy in Vehicular Social Networks

Yuexia Zhang, *Member, IEEE*, Ying Zhou, Siyu Zhang, *Member, IEEE*, Guan Gui, *Senior Member, IEEE*, Bamidele Adebisi, *Senior Member, IEEE*, Haris Gacanin, *Fellow, IEEE*, and Hikmet Sari, *Life Fellow, IEEE*

***Abstract*—Limited edge server resources and uneven distribution of traffic density in vehicular networks result in problems such as unbalanced network load and high task processing latency. To address these issues, we proposed an efficient caching and offloading resource allocation (ECORA) strategy in vehicular social networks. First, to improve the utilization of vehicular idle resources, a collaborative computation and storage resource allocation mechanism was designed using mobile social similarity. Next, with the optimization objective of minimizing the average task processing delay, we studied the combined resource allocation optimization problem and decoupled it into two subproblems. For the service caching subproblem, we designed a stable matching algorithm by mobile social connections to dynamically update the cache resource allocation scheme for improving the task unloading efficiency. For the task offloading subproblem, a discrete cuckoo search algorithm based on differential evolution was designed to adaptively select the best task offloading scheme, which minimized the average task processing delay. Simulation results revealed that the ECORA strategy outperformed the resource allocation strategy based on particle swarm optimization and genetic algorithm, and reduced the average task processing delay by at least 7.59%. Meanwhile, the ECORA strategy can achieve superior network load balancing.**

***Index Terms*—Cuckoo algorithm, resource allocation, task offloading, vehicular social networks.**

I. Introduction

With the rapid development of 5G communication technologies, novel compute-intensive and latency-sensitive mobile vehicular services have attracted considerable research attention for applications such as remote driving, environmental awareness, and 4K live streaming [1]–[5]. Mobile Edge Computing (MEC) is used in the Internet of vehicles can reduce the transmission distance, improve user experience, and enhance resource allocation efficiency [6], [7]. Therefore, studying the resource allocation strategy for vehicular networks is crucial [8]–[11]. However, considering the rapid increase in the number of intelligent connected vehicles and data traffic in the future, the limited edge server resources render timely response to the massive task requests from vehicle users difficult. Therefore, the resource allocation efficiency of edge server is reduced, and task processing delay is considerably affected, which renders the design of the resource allocation algorithm challenging [12]–[16].

Recently, multi-edge server collaboration have attracted considerable research attention [17]. In this technique, multiple edge servers collaborate to perform computing tasks, including scheduling storage and computing resources, to maximize network performance and resource utilization [18]–[20]. Based on superior edge server collaborations, a new cooperation system should be considered to integrate and use the idle computing resources in intelligent networked vehicles [21]. Thus, the original available resources in the server collaboration system should be extended to optimize the quality of the user experience. According to the global intelligent connected vehicle forecast report (2020-2024) released by International Data Corporation (IDC), the global shipments of intelligent networked vehicles will reach approximately 76.2 million units by 2024 [22]. More than 71% of new cars shipped globally will be equipped with smart connectivity systems. Full use of the idle resources of a large number of intelligent networked vehicles will relieve the resource shortage of the Internet of vehicles, enhance the scalability of the computing services, and improve the service quality of vehicle users.

However, research on edge server and vehicle collaborative resource allocation has some key problems. First, in most studies, only the task unloading process between mission vehicles, edge servers, and collaborative vehicles in the Internet of vehicles is analyzed, and the effect of the collaborative vehicle service caching process on task unloading is ignored. Specifically, cooperative vehicles exhibit sufficient computing resources. However, if the cooperative vehicles do not exhibit cache-related services, it cannot perform corresponding computing tasks. Thus, this phenomenon leads to the inadequate utilization of the computing resources of cooperative vehicles. Second, because of the limited storage resources of collaborative vehicles, only a few services can be cached at a

The work was supported in part by Sub Project of National Key Research and Development plan in 2020 under Grant No. 2020YFC1511704, Beijing Information Science and Technology University (No.2020KYNH212, No.2021CGZH302). Beijing Science and Technology Project under Grant No. Z211100004421009, and in part by the National Natural Science Foundation of China under Grant No. 61971048.

Y. Zhang is with Key Laboratory of Information and Communication Systems, Ministry of Information Industry, and Laboratory of Optoelectronic Measurement Technology and Instrument, Ministry of Education, Beijing Information Science and Technology University, Beijing 100101, China (e-mail: zhangyuexia@bistu.edu.cn).

Y. Zhou, and S. Zhang are with Key Laboratory of Modern Measurement and Control Technology, Ministry of Education, and School of Information and Communication Engineering, Beijing Information Science and Technology University, Beijing 100101, China (e-mail: zhang1fs@bistu.edu.cn).

G. Gui and H. Sari are with the College of Telecommunications and Information Engineerning, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: guiguan@njuui.edu.cn, hsari@ieee.org).

B. Adebisi is with the Department of Engineering, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom. (E-mail: b.adebisi@mmu.ac.uk).

H. Gacanin is with the Faculty of Electrical Engineering and Information Technology, RWTH Aachen University, Aachen 55-52062, Germany (e-mail: harisg@ieee.org).

time. Considering the task between vehicle unloading request is random and the inability to perform advance prediction, the current allocation of resources process should be overhauled to incorporate an intelligent caching mechanism to perceive mobile social connections between the vehicles. Finally, in the Internet of vehicles, the task load and resource status of various edge servers change with the vehicle flow density, speed, and other factors. Therefore, the influence of vehicle movement can be attributed to load balancing among edge servers should be considered.

This study proposed an efficient caching and offloading resource allocation (ECORA) strategy in vehicular social networks. Unlike existing studies, this study focuses on resource allocation of joint service caching and task offloading between vehicles and mobile edge servers, in the cooperative communication scenario. To realize edge server load balancing and improve the utilization rate of idle resources of intelligent networked vehicles, we comprehensively analyzed the influence of social contact and motion state between vehicles on service caching and task unloading process. We propose a stable matching algorithm based on mobile social contact and a discrete cuckoo search algorithm based on differential evolution. This strategy supports dynamic update and allocation of cache resources and assists the vehicle to adaptively select the best task unloading location. Thus, the service quality can be improved, and the average task processing delay can be reduced. The main contributions are summarized as follows.

- A social vehicle communication system model combining service caching and task unloading was designed. By using multi-edge servers and vehicle cooperation to allocate resources, it realized network load balancing and optimized the efficiency of idle resources allocation.
- An efficient caching and offloading resource allocation strategy in vehicular social networks was proposed. In this strategy, the average task processing delay is minimized for achieving high service quality and the NP-hard resource allocation problem of joint service cache and task unload are categorized into two sub-problems of service cache and task unload. To address the subproblem of service cache, a stable matching algorithm based on mobile social contact was designed. To address the task unloading subproblem, a discrete cuckoo search algorithm based on differential evolution was designed.
- Our results revealed that compared with the particle swarm optimization (PSO) [23] and genetic algorithm (GA) strategies [24], the ECORA strategy reduced the average task processing delay by at least 7.59% and 9.98% with improved network load balancing.

II. Related Work

Considering that the limited resources of conventional edge servers cannot timely process the computing task requests of numerous vehicle users, the collaborative resource allocation of multiple edge servers has attracted considerable research attentions [25]–[28]. For example, Zhu *et al*. [29] proposed a collaborative resource allocation strategy for multiple edge servers. Specifically, in this strategy, the total energy constraint and individual energy constraint of each server as well as the limitation of individual computing frequency are considered and when to unload the corresponding task to the specified server is determined. Thus, the task processing delay can be minimized. Guo *et al* [30] proposed an uninstallation strategy based on game theory for multi-edge server cooperation scenarios. By studying the stable balance among computation cost, energy consumption and delay, they reduced energy consumption and processing delay. Mao et al. [31] proposed a task unloading mechanism for Internet of Vehicles based on trusted RSU services, built a new infrastructure trust management model. According to introduce social factors to strengthen the security management of RSU, it can improve the system's ability to handle task unloading when attacked and maintain low delay and high task success rate. Furthermore, Wang et al. [32] proposed a collaborative resource allocation strategy between edge cloud and central cloud. They classified tasks into according to priorities based on delay tolerance and subsequently reordered the tasks. The reinforcement learning algorithm was used to intelligently allocate local edge server resources and cloud resources, thus optimizing the service quality of vehicle users. Yin et al. [33] designed a UAV assisted multi-input multi-output non-orthogonal multi-access (MIMO-NOMA) resource allocation strategy based on the wireless caching network, and optimized the UAV deployment scheme, hybrid beamforming scheme and power allocation scheme to achieve lower user delay during content delivery. Many studies have focused on maximizing the use of computing resources by building edge server clusters to reduce computing overhead and latency [34]–[36]. However, in the aforementioned study, the allocation of resources of edge servers is considered, and the computing and storage resources of intelligent networked vehicles are ignored. Moreover, as the number of vehicles and requests increases, the limited resources of edge servers cannot satisfy the service user requirement.

Due to the increasing demand of vehicular networks for resources related to computing and caching, in this case, each vehicle has different types of resources, and these resources can be shared independently, how to flexibly allocate resources among multiple vehicles is challenging. Pradhan *et al*. [37] proposed a semi-Markovian decision based resource allocation mechanism to manage resources from different vehicles and allocate necessary resources to their users on demand to improve resource management revenue. Kim *et al*. [38] proposed an optimal job partitioning and allocating algorithm for vehicular cloud computing, which minimized the overall execution time of jobs by tracking available resources, analyzing the optimal number and size of task division. Considering the high mobility of vehicles, when vehicles frequently join or leave the mobile network, the risk of communication link failure and communication overhead will be greatly increased, and the efficiency of resource sharing will be reduced. Gu et al. [39] proposed a novel three-stage joint resource allocation and RIS optimization algorithm. Considering channel quality and social trust between V2X links, they established a RIS assisted vehicular network communication system with social trust. Thus, solving RIS reflection coefficient design, power allocation of each pair of vehicles and spectrum reuse mode

optimization in stages. Huang *et al*. [40] proposed a task offloading and resource allocation strategy based on joint task type and speed perception. Considering that different types of tasks and speed need different delay requirements, further analyzed the internal relationship among speed, task type and delay requirements, and established a joint task type and speed perception delay constraint model. Enables more accurate task offloading and resource allocation. The above research focuses on the consideration of multi-vehicle cooperation to assist the vehicular networks to carry out the task unloading process, which can not effectively take advantage of the rich resources of MEC. If the vehicle requests to unload too many tasks, the resources of the edge server can not work together to execute the tasks, the sum of the computing requirements of these tasks may exceed the total computing resources of the multi-vehicle collaborative system. Such problems will lead to the overload of multi-vehicle cooperative system and the deterioration of QoS of tasks.

Numerous studies have been conducted to alleviate the limited resources of the Internet of vehicles, enhance vehicular networks computing service scalability as well as computing power. In the current research, multi-edge server and multi-vehicles can cooperate to allocate resources using idle resources of vehicles [23], [24], [41], [42]. Thus, effectively satisfying the new generation vehicle computational and delay sensitive business service requirements is critical. For example, Li *et al* [24] proposed a novel resource allocation strategy based on GA in multi-user and multi-edge server scenarios. In this strategy, the interaction between unloading decision and resource allocation is analyzed comprehensively, and reduce the energy consumption in the unloading decision process by using the GA to constrain the channel selection and power allocation. Hou *et al*. [23] proposed a resource allocation strategy based on PSO for cooperation between mobile vehicles and edge computing nodes. The strategy describes the effect of compute node selection on transmission link reliability during uninstallation and task assignment. Xu et al. [43] proposed a short-term resource allocation strategy for content provider (CP) and content requester (CR), which used the contract theory to allocate communication and computing resources for each potential CP-Cr pair and matched CP and CR based on a stable matching algorithm. Thus, realizing efficient content sharing between vehicles or between vehicles and infrastructure. A PSO algorithm was designed to maximize transmission link reliability under delay constraints. A reprocessing strategy was introduced to prevent possible task computing interruption and failure, which enhances the fault tolerance of system task processing and optimizes user service experience. To alleviate the shortage of spectrum resources, vehicles can not only communicate through the licensed spectrum, but also consider offloading part of the computing tasks to the edge server through the unlicensed spectrum. Furthermore, to address the uncertainty of vehicle movement and improve service reliability, a task replication strategy was proposed to allow multiple cooperative vehicles to perform a task simultaneously [37], [39], [44]. The aforementioned studies have focused on the task unloading process in the scenario of cooperation between vehicles and edge servers, without considering the effect of the service caching process on resource allocation. The service caching process is combined with the task unloading process, and the cooperative vehicle cannot perform corresponding computing tasks without caching related services. Moreover, most of the work ignores the mobile social connection between the cooperative vehicle and the mission vehicle. For example, the initiative of the cooperative vehicle's service cache and its willingness to assist the unloading task are not considered. In this case, the cooperative vehicle has a high probability of rejecting the unloading request and the communication link established is short in duration. This will lead to the reduction of task unloading efficiency and the decline of vehicle service quality. Therefore, it is critical to joint optimize the task unload and service caching.

III. System Model and Definitions

In this study, we proposed an efficient caching and offloading resource allocation strategy in vehicular social networks. First, we establish a social vehicular communication system model combining service caching and task unloading (Fig. 1). As displayed in Fig. 1(a), the model contains two layers, namely the physical and mobile social layers. The physical layer consists of several roadside units (RSU), edge servers, mission vehicles, and collaborative vehicles, which can establish communication, service caching, and task unloading. The mobile social layer is mapped by the physical layer and used to determine willingness based on social relations, mobile relevance, and resource idle rate. Thus, willingness of the vehicles to collaborate on resource allocation, service caching and task offloading, as well as between vehicles and edge servers is considered.

As displayed in Fig.1.(b), the model consists of M mission vehicles, N collaboration vehicles, and K roadside units (RSUs), and edge servers deployed at the RSUs. Here, the collection of mission vehicles is referred to $MV = \{MV_1, MV_2, ..., MV_m, ..., MV_M\}$, which is used for unloading tasks to RSUs (or cooperative vehicles) within the communication range. A collection of collaborative vehicles, known as $CV = \{CV_1, CV_2, ..., CV_n, ..., CV_N\}$, provides computing resources to complete the tasks of mission vehicles. In this study, RSUs and edge servers configured around them are regarded as a whole, denoted as access points (APs), and their set is represented by $AP = \{AP_1, AP_2, ..., AP_k, ..., AP_K\}$. The AP has lightweight computing and caching resources that place the caching service on the collaborative vehicle and provide computing resources to complete the tasks of the mission vehicle. Furthermore, the total local storage resource size of MV_m is set to S_m, and the total local computing resource size is set to C_m. The local total storage resource size of collaborative vehicle CV_n is defined as S_n, and the local total computing resource size is defined as C_n. The total local storage resource size of AP_k is defined as S_k, and the total local computing resource size as C_k.

In the mobile social layer, this layer consists of the physical layer mapping, as well as multi-mission vehicles, collaborative vehicles, RSUs, and edge servers deployed at the RSUs. Among them, the mission vehicle can determine whether the

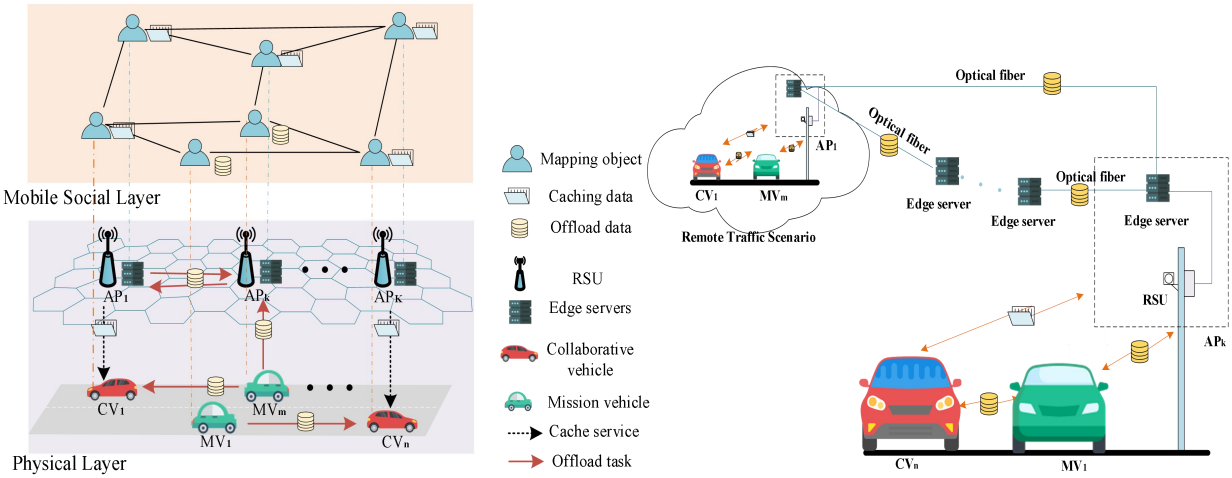


Fig. 1. ECORA system model.(a).social vehicular communication system. (b).Physical layer of social vehicular networks communication system.

cooperative vehicles and RSUs can unload the task and return the result according to the mobile social relationship. RSU can determine whether collaborative vehicles are willing to cache service data and related applications according to mobile social relations. Furthermore, RSU can determine whether edge servers of other RSUs are willing to cooperate to complete computing tasks. The system model can provide F service cache resources, and SR_f, $f \in \{1, 2, ..., F\}$ represents the size of the f kinds of service caching resources.

In this study, the total bandwidth of the available spectrum in the system model is assumed to be W, which is categorized into several orthogonal sub-channels, and the bandwidth of each subchannel is B_0. Each V2V link can reuse one V2I subchannel resource for data transmission. All service APs and cooperative vehicles are assumed to constitute a set SD, and $SD = \{SD_1, SD_2, ..., SD_y, ...\}$, $Y \in \{CV \cup AP\}$, $y \in Y$, where SD_y represents the y^{th} collaborative vehicle (or access point) in the set that provides the service. Tasks generated by the MV_m of the mission vehicle can be unloaded to cooperative vehicle CV_n, associated access point AP_k, and cooperative access point $AP_{k'}$. Therefore, studying the MV_m unloading task process based on three situations is critical.

A. Service Caching Mechanism

When the access point and the collaborative vehicle provide services to the mission vehicle, the corresponding services should be cached locally in advance. Because of the limited cache resources of collaborative vehicles, all services cannot be cached locally simultaneously. Therefore, at the beginning of each time slot, the access point determines the services the collaborative vehicle should cache based on popularity. Here, popularity ρ is defined as the popularity of different contents within a certain interaction time, following Zipf distribution and expressed as follows:

$$p(f) = \frac{f^{-\alpha}}{\sum_{f=1}^{F} f^{-\alpha}}, \ f \in F, \tag{1}$$

where parameter α describes the steepness of the Zipf distribution and reflects the popularity of various contents. Assuming that $S^t = \{S_y^t, y \in Y\}$ represents the service caching policy of all cache devices in the t slot, and considering that the access point can directly obtain the cache content from the core network, and the transmission time is ignored. Here, $S_{y,y\in AP}^t = 1$ means all APs in the t slot can provide all cache resource services. Furthermore, $S_{y,y\in CV}^t$ represents the service caching strategy of t slot cooperative vehicles.

$S_{y,f}^t \in \{0, 1\}$, $y \in CV$ is defined to represent the CV_y cache resource allocation strategy in the t slot. For representation, the variable t representing the time slot is ignored for caching and unloading the following policies. Furthermore, $S_{y,f} = 1$ indicates that the f^{th} service resource was cached by cooperative vehicle CV_y in the t time slot, whereas $S_{y,f} = 0$ denotes that the f^{th} service resource was not been cached by cooperative vehicle CV_y in the t time slot. When CV_y caches service resources, it cannot violate its own cache capacity limit $\mathbb{Z}_y$. Therefore, the constraint conditions for collaborative vehicle CV_y to cache service resources are as follows:

$$\sum_{f=1}^{F} S_{y,f} \cdot SR_f \leq \mathbb{Z}_y, \forall y \in CV. \tag{2}$$

However, the popularity of service resources can change over time. Therefore, cooperative vehicles consider whether to replace cached content at each time slot. When a new data packet arrives, the cooperative vehicle determines whether its remaining cache capacity is greater than the packet capacity. If the remaining cache capacity is sufficient, the data packet is saved. Otherwise, the existing cache services are sorted quantitatively by popularity, and the cache services with low popularity are deleted in batches to replace the new data packets with high popularity.

Furthermore, to avoid overloading the access point and ensure CV_n can determine whether the desired service cache data can be downloaded from AP_k, we introduced an $N_K \times N_N$-dimensional matrix B. Here, $b_{k,n}$ represent the elements in the

k^{th} row and n^{th} columns of the matrix. Furthermore, $b_{k,n} = 1$ indicates that collaborative vehicle CV_n can communicate with AP_k and download service resources, whereas $b_{k,n} = 0$ denotes that collaborative vehicle CV_n cannot communicate with AP_k and download service resources. Because of the limited load of the AP_k, the AP_k cannot violate its own maximum connection number limit Λ when providing cache resources. Therefore, the AP_k should satisfy the following constraints when providing service cache:

$$\sum_{n=1}^{N} b_{k,n} \leq \Lambda. \tag{3}$$

B. Task Offloading Mechanism

Each mission vehicle is assumed to generate a computational task in each time slot, and the mission vehicle prioritizes unloading the task to the cooperative vehicle with the best channel quality within the communication range. Furthermore, the time required for the access point to download the cache service of the core network through optical fiber is ignored, and the default access point can cache all services. If the cooperative vehicle associated with the mission vehicle has insufficient resources or does not cache the required services, the mission vehicle offloads the computing task to the access point. Furthermore, if the access point associated with the mission vehicle has insufficient resources, the access point unloads the computing task to another access point. Therefore, at the beginning of each time slot, the mission vehicle should determine the unloading position of the task according to the idle rate of resources, social relations, and mobility relevance.

First, the idle rate of computing resources is defined as ρ_y^f, which represents the proportion of idle computing resources to the total computing resources in SD_y, that is, $\rho_y^f = \rho_y^e / C_y$, $(y \in Y)$. Here, ρ_y^e indicates the size of idle computing resources of SD_y, C_y indicates the total local computing resource size of the SD_y.

Here, $A_{m,y}$ is defined as the computing resource allocation strategy, which is used to indicate whether the mission vehicle MV_m unloads the task at SD_y. Furthermore, $A_{m,y} = 0$ indicates that MV_m is not unloaded from SD_y, whereas $A_{m,y} = 1$ indicates that MV_m unloads tasks from SD_y. In each time slot, mission vehicle MV_m can select only one unloading position to perform the unloading task. Therefore, constraint conditions should be satisfied as follows:

$$\sum_{y=1}^{Y} A_{m,y} = \sum_{n=1}^{N} A_{m,n} + \sum_{k=1}^{K} A_{m,k} + \sum_{k=1}^{K} \sum_{k'=1,k' \neq k}^{K} A_{m,k,k'} = 1 \tag{4}$$

where $A_{m,n}$ is a binary variable, which is used to indicate whether MV_m unloads the task at CV_n, $A_{m,k}$ is a binary variable used to indicate whether MV_m unloads the task at AP_k, $A_{m,k,k'}$ is a binary variable used to indicate whether MV_m unloads the task at $AP_{k'}$ associated with AP_k.

To ensure that the mission vehicle preferentially selects the cooperative vehicle (or access point) associated with itself to unload the task, the idle rate threshold ρ_{th}^f is set. Thus, the times of cooperation and related costs are reduced considerably. At this stage, the constraint conditions for the mission vehicle to select the cooperative vehicle (or access point) to unload the task are denoted as (5), where if a cooperative vehicle computing resource idle rate ρ_y^f is greater than the given threshold ρ_{th}^f, and the cooperative vehicle has cached the corresponding service content. Next, MV_m selects the unload task of the cooperative vehicle. If the ρ_y^f of any cooperative vehicle is lower than the specified threshold, and the ρ_y^f of the access point associated with MV_m is greater than the specified threshold, MV_m unloads the task at the access point.

Furthermore, if the computing resource idle rate ρ_y^f of any cooperative vehicle is lower than the specified threshold, the computing resource idle rate ρ_y^f of the access point associated with MV_m is lower than the specified threshold, and the computing resource idle rate ρ_y^f of an access point is greater than the specified threshold ρ_{th}^f. Then MV_m selects the access point to unload the task. If none of the aforementioned conditions is satisfied, MV_m stops the unloading task request and waits for the emergence of the cooperative vehicle (or access point) SD_y that meets the unloading requirements in the next time slot.

However, in vehicular networks, multiple SD_y that satisfy the requirements exist when mission vehicle MV_m chooses unloading task according to constraints. At this stage, MV_m can score and sort cooperative vehicles (or APs) that satisfy the requirements according to social relationship, mobility association, and idle rate of computing resources. Then MV_m selects high-quality cooperative vehicles (or APs) with high social mobility similarity to unload tasks.

Furthermore, the social relationship between MV_m and cooperative vehicle (or AP) SD_y can be expressed by interest similarity and social trust degree. The similarity of interest can be defined as the social content similarity of historical browsing between MV_m and SD_y (the connection duration of interaction service and the number of interaction service resource). Users with high interest similarity are often more likely to share data in the future. Let $\overrightarrow{I_a^f} = (I_{a1}^f, I_{a2}^f)$ be the interest feature vector of the users, where $\overrightarrow{I_m^f}$ represents the interest feature vector of the MV_m, $\overrightarrow{I_y^f}$ represents the interest feature vector of the SD_y. Cosine similarity is used to measure the interest similarity $S_{m,y}$ between SD_y and MV_m, which is expressed as follows:

$$S_{m,y} = \frac{\sum_{f=1}^{F} (\overrightarrow{I_m^f} \cdot \overrightarrow{I_y^f})}{\sum_{f=1}^{F} (||\overrightarrow{I_m^f}|| \times ||\overrightarrow{I_y^f}||)}. \tag{6}$$

We used the intermediary centrality to measure the degree of social trust between MV_m and SD_y. Here, dots represented MV_m and SD_y. Edges represented the social connection between MV_m and SD_y. Therefore, we can develop mobile social relationship networks and calculate the intermediary center based on this network. The degree of social trust between MV_m and SD_y can be expressed as follows:

$$B_{m,y} = \sum_{m,y \in V} \frac{g_{m,y}(e)}{G_{m,y}}, \tag{7}$$

where $G_{m,y}$ represents the number of shortest paths between MV_m and SD_y, $g_{m,y}(e)$ represents the number of paths

$$\begin{cases} A_{m,y}=1, & \{p_y^f \geq p_{th}^f, y \in CV\} \cup \{S_{y,f}=1\} \\ A_{m,y}=1, & \{p_y^f < p_{th}^f, y \in CV\} \cup \{p_y^f \geq p_{th}^f, y \in AP \text{ and associated with (a.w.) } MV_m\} \\ A_{m,y}=1, & \{p_y^f < p_{th}^f, y \in CV\} \cup \{p_y^f < p_{th}^f, y \in AP \text{ and a.w. } MV_m\} \cup \{p_y^f \geq p_{th}^f, y \in AP \text{ and not a.w. } MV_m\} \\ A_{m,y}=0, & \text{otherwise} \end{cases} \tag{5}$$

containing edges with a weight of e, and V represents the number of points. To facilitate calculation, the normalized social trust can be expressed as follows:

$$\overline{B_{m,y}} = \frac{B_{m,y}}{(K+N+M)^2}. \tag{8}$$

Considering the interest similarity and social trust between MV_m and SD_y, the social connection between MV_m and SD_y can be expressed as follows:

$$\theta_{m,y} = \alpha_2 S_{m,y} + \beta_2 \overline{B_{m,y}}, \tag{9}$$

where α_2 and β_2 are debug factors, satisfying $\alpha_2 = \beta_2 = 0.5$.

Finally, we introduce movement correlation and study its effect on task unloading. In mobility relevance, each vehicle change their location and driving direction in real time, which is the topological mobility in the Internet of vehicles. Therefore, the position change between MV_m and SD_y, and the change of MV_m's own driving direction should be considered comprehensively to determine whether MV_m and SD_y are willing to unload tasks. For the whole Internet of vehicles, the process of task unloading should be scattered in various network topologies of the Internet of vehicles as much as possible, rather than concentrated in a certain area. This method can reduce the load pressure of RSU and edge servers and the mission vehicle can unload data as close as possible. According to vehicle mobility, we establish the position correlation function $D_{m,y}$:

$$D_{m,y} = 1 - e^{(-\frac{\mu \cdot R_y}{d_{m,y}})}, \tag{10}$$

where $d_{m,y}$ is the Euclidean distance between MV_m and SD_y, and R_y is the communication radius of SD_y. Here, μ is the weight parameter, and its expression as follows:

$$\mu = \begin{cases} 1, & MV_m \text{ drives to } SD_y \\ 0.5, & MV_m \text{ leaves to } SD_y \end{cases}, \tag{11}$$

where $\mu = 1$ indicates that when MV_m drives to SD_y, the connection between MV_m and SD_y lasts for a long time. Then MV_m and SD_y are highly likely to remain connected in the future for some time, and the probability of unloading task increases. Here, $\mu = 0.5$ indicates that when MV_m drives away from SD_y, the connection between MV_m and SD_y lasts for a short time, and the probability of unloading tasks decreases and data interaction increases. At this stage, tasks should be scattered in the network topology rather than centralized so that MV_m will unload tasks in a further place.

C. Optimization

When mission vehicle MV_m generates task T_m and unloads it to the collaborative vehicle CV_n for execution, the execution delay can be categorized into two parts: 1) The transmission delay $t_{m,n}^{up}$ of MV_m uploading data to CV_n. 2) The calculated delay $t_{m,n}^{ul}$ required for CV_n to perform the task T_m, that is $t_{m,n}^{ul} = c_m / c_{m,n}$. Here, c_m represents the MV_m's computing amount of the task, $c_{m,n}$ represents the computing resources that CV_n allocates to task T_m. Task T_m generated by MV_m is unloaded to CV_n, and the time delay required for task execution is $t_{m,n}^{p}$ as follows:

$$t_{m,n}^{p} = t_{m,n}^{up} + t_{m,n}^{ul}, \tag{12}$$

where MV_m generates T_m and unloads it to associated AP_k for execution. In this case, the execution delay includes two parts: 1) The transmission delay of MV_m uploading data to AP_k. 2) The calculated delay $t_{m,k}^{ul}$ required for AP_k to perform the task. Here, $c_{m,k}$ represents the computing resources AP_k allocates to a task. The delay required for task T_m generated by MV_m to unload to AP_k associated with itself to perform the task is $t_{m,k}^{p}$ and expressed as follows:

$$t_{m,k}^{p} = t_{m,k}^{up} + t_{m,k}^{ul}. \tag{13}$$

After MV_m generates T_m and unloads it to associated AP_k for execution, the AP_k may not perform task because of insufficient computing resources. In this case, AP_k cooperates with other surrounding APs to fully utilize idle computing resources. Thus, system resource utilization and enhancing user experience are improved. If AP_k selects an idle access point $AP_{k'}$ to cooperate in task execution, then the execution delay can be categorized into three parts: 1) The transmission delay of MV_m uploading data to AP_k. 2) The transmission delay of AP_k uploading data to $AP_{k'}$. 3) The calculated delay $t_{m,k'}^{ul}$ required by $AP_{k'}$ to perform task T_m, that is, $t_{m,k'}^{ul} = c_m / c_{m,k'}$. $c_{m,k'}$ represents the computing resources $AP_{k'}$ allocates to a task. The time delay required when the task T_m generated by MV_m is unloaded from $AP_{k'}$ to perform the task is $t_{m,k,k'}^{p}$, and it is expressed as follows:

$$t_{m,k,k'}^{p} = t_{m,k}^{up} + t_{k,k'}^{s} + t_{m,k'}^{ul}. \tag{14}$$

In mission vehicle requesting task unloading, the objective function of the optimization problem can be expressed as

follows:

$$
\begin{aligned}
T = & \\
& \min \sum_{y=1,y\in Y}^{N} \sum_{n=1}^{M} \sum_{m=1}^{K} \sum_{k=1}^{K} \sum_{k'=1,k'\neq k} \frac{A_{m,y}(t^p_{m,n}+t^p_{m,k}+t^p_{m,k,k'})}{M} \\
& s.t.\, C_1:\ QoS_{m,y} \geq \psi, \forall m \in MV, y \in Y \\
& \quad C_2:\ \sum_{y=1} A_{m,y} = 1, \forall m \in MV, y \in Y \\
& \quad C_3:\ \sum_{f=1}^{F} S_{y,f} \cdot SR_f \leq \mathbb{Z}, \forall y \in CV \\
& \quad C_4:\ \sum_{n=1}^{N} b_{k,n} \leq \Lambda, \sum_{k=1}^{K} b_{k,n} \leq 1 \\
& \quad C_5:\ 0 \leq c_{m,n} \leq C_n, \sum_{m=1}^{M} c_{m,n} \leq C_n, \forall n \in CV \\
& \qquad\ \ 0 \leq c_{m,k} \leq C_k, \sum_{m=1}^{M} c_{m,k} \leq C_k, \forall k \in AP \\
& \qquad\ \ 0 \leq c_{m,k'} \leq C_{k'}, \sum_{m=1}^{M} c_{m,k'} \leq C_{k'}, \forall k' \in AP
\end{aligned} \tag{15}
$$

where $A_{m,y}$ is defined as the computing resource allocation strategy, which is used to indicate whether the mission vehicle MV_m unloads the task at SD_y. Constraint C_1 indicates that to ensure the user experience of mission vehicles, the $QoS_{m,y}$ of communication links is required to exceed the minimum QoS threshold ψ. The constraint C_2 indicates that the mission vehicle can select only one unloading position to perform the unloading task. The constraint C_3 indicates that, considering the limited cache resources, to prevent the waste of cache resources, limiting the cached resources of a CV is necessary while ensuring that the cache space occupied by any CV does not exceed $\mathbb{Z}$. Constraint C_4 indicates that when an access point AP_k provides a cache resource service, its maximum connection number does not exceed Λ, and the cooperative vehicle can only select one access point to receive the service cache. Constraint condition C_5 indicates that the size of computing resources allocated to the mission vehicle is limited, while ensuring that the total computing resources allocated to the mission vehicle cannot exceed the maximum value provided by the cooperative vehicle (or access point).

IV. Our Proposed ECORA Strategy

Because the original optimization problem is NP-hard, obtaining the optimal joint cache and computational resource allocation strategy in the polynomial time is critical. To simplify the problem, the combinatorial optimization problem was categorized into two sub-problems, namely service caching subproblem and task unloading subproblem.

A. Stable Matching Algorithm Based on Mobile Social Contact

To address the service cache subproblem. First, the service cache requirement relationship between AP_k and CV_n was modeled as a matching model. In t time slot, CV_n requests f^{th} service cache resource. For AP_k, the data transmission is assumed to complete in a short time with a mission vehicle with high social similarity to the service under the QoS requirements of the requesting business. Therefore, the effect function of AP_k on CV_n is expressed as follows:

$$
Y^{AP}_{k,n} = \frac{\theta_{k,n} \cdot QoS_{k,n}}{t^{up}_{k,n}}, \tag{16}
$$

where $\theta_{k,n}$ represents the social connection between CV_n and AP_k, and $\theta_{k,n} = \alpha_2 S_{k,n} + \beta_2 \overline{B_{k,n}}$ can be obtained. At this stage, $S_{k,n}$ is the interest similarity between CV_n and AP_k, and $\overline{B_{k,n}}$ is the normalized social trust. $QoS_{k,n}$ represents the communication service quality between CV_n and AP_k, and $QoS_{k,n} = \alpha_1 P_{k,n} + \beta_1 QR_{k,n}$ can be obtained, where α_1 and β_1 are debug factors, satisfying $\alpha_1 = \beta_1 = 0.5$. Then $QR_{k,n}$ is the link state between CV_n and AP_k.

Unlike AP_k, CV_n mainly considers its relative position with the access point and the popularity of the cache content of the request service. CV_n also considers whether a communication link exists between AP_k and CV_n. Based on this, the effect function of CV_n on AP_k is expressed as follows:

$$
Y^{CV}_{k,n} = \frac{D_{k,n} \cdot QR_{k,n}}{1 + e^{-\rho(f)max}}, \tag{17}
$$

where $D_{k,n}$ represents the moving correlation between CV_n on AP_k, and $D_{k,n} = 1 - \exp(-\mu R_k / d_{k,n})$ can be obtained. In this case, R_k is the communication radius of AP_k, and $d_{k,n}$ is the Euclidean distance between CV_n on AP_k. Here, $\rho(f)_{max}$ indicates that CV_n selects the cache service with the highest popularity from its uncached service set and requests sharing to AP_k.

The order of CV in AP_k's preference list PR^{AP_k} is mainly in descending order according to the size of $Y^{AP}_{k,n}$, that is, the CV_n corresponding to the maximum value of $Y^{AP}_{k,n}$ ranks first in the list of PR^{AP_k}. Similarly, AP_k in CV's preference list PR^{CV_n} is arranged in the descending order according to $Y^{CV}_{k,n}$. Based on the aforementioned assumptions, one-to-one stable matching is defined as follows:

B. Discrete Cuckoo Search Algorithm Based on Differential Evolution

Based on the aforementioned algorithm, the system iterates continuously until no blocking pair exists in the matching results of service cache. Thus, the service caching subproblem is solved, which improves the efficiency of collaborative vehicle service cache and boosts the success rate of task unloading. Moreover, we proposed a discrete cuckoo search algorithm based on differential evolution to solve the task unloading subproblem.

The position of the i^{th} nest in the t generation is assumed to be x^t_i, and $x^t_i = \{x^t_{i,1}, x^t_{i,2}, ..., x^t_{i,m}, ..., x^t_{i,M}\}$, where M represents the dimension. Furthermore, x^{t+1}_i represents the new location of the i^{th} nest after global update of x^t_i of the nest location of the t generation. At this stage, the path and

Algorithm 1 Stable matching algorithm based on mobile social contact.

Each $AP_{k^{**}}$ and $CV_{n^{**}}$ establishes their own initial preference lists $PR^{AP_{k^{**}}}$ and $PR^{CV_{n^{**}}}$, and records the unmatched $CV_n : DM = \{CV_n, \forall CV_n \in CV\}$

while $DM \neq \emptyset$, $\exists PR^{CV_{n^{**}}} \neq \emptyset$ **do**:
  if $\sum_{f=1}^{F} S_{n,f} \cdot SR_f \leq \mathbb{Z}_n$ **then**:
    CV_n in turn sends data caching requests to the AP_k with the highest order in the preference list;
  else
    CV_n deletes the least popular local service cache content;
    CV_n continues to send data cache requests to the AP_k with the highest order in the preference list.
  end if
  for $\forall AP_k \in AP$, $1 \leq k \leq K$ **do**:
    if AP_k prefers CV_n to CV_{n^*}, which matches the last row of the candidate list, and meets $\sum_{n=1}^{N} b_{k,n} < \Lambda$ **then**:
      AP_k temporarily accepts the CV_n and reorders the preference list;
      CV_n moves out of DM;
    else if AP_k prefers CV_n to CV_{n^*} , which matches the last row of the candidate list, and meets $\sum_{n=1}^{N} b_{k,n} = \Lambda$ **then**:
      AP_k temporarily accepts CV_n and reorders the preference list;
      AP_k removes CV_{n^*} from its preference list;
      CV_{n^*} removes AP_k from its preference list;
      CV_{n^*} is moved into DM and CV_n is moved out of DM.
    else if AP_k does not prefer CV_n compared with CV_{n^*} that matches the last row of the candidate list **then**:
      AP_k refuses CV_n, CV_n removes AP_k from its preference list;
    end if
  end for
end while

position of cuckoo searching for parasitic nests are updated as follows:

$$x_{i,m}^{t+1} = x_{i,m}^{t} + \alpha_3 \cdot Rand \cdot Levy(\beta_3),\ i = 1, 2, \cdots, K+N, \tag{18}$$

where $x_{i,m}^{t}$ represents the value of the i^{th} nest in the m dimension of the nest position in the t generation. Similarly, $x_{i,m}^{t+1}$ is the value of the m dimension in the new position of the i^{th} nest. Here, α_3 is the step size factor and used to control the step size. This value is typically set to be $\alpha_3 = 1$. Where $Levy(\beta_3) \sim u = t^{-\beta_3}$, $(1 < \beta_3 \leq 3)$, β_3 is the influence factor, typically $\beta_3 = 1.5$. The $Levy$ distribution is expressed as follows:

$$Levy(\beta_3) = 0.01 \cdot \frac{u}{|v|^{\frac{1}{\beta_3}}} \cdot (x_{j,m}^{t} - b_{g,m}^{t}),\ j, g = 1, 2, \cdots, K+N, \tag{19}$$

where u and v both obey normal distribution, that is, $u \sim N(0, \sigma_u^2)$, $v \sim N(0, \sigma_v^2)$. $\sigma_u = \left\{ \frac{\Gamma(1+\beta_3) \cdot \sin(\pi \cdot \beta_3/2)}{\Gamma[(1+\beta_3)/2] \cdot \beta_3 \cdot 2^{(\beta_3-1)/2}} \right\}^{1/\beta_3}$, $\sigma_v = 1$. In addition, $b_g^t = \{b_{g,1}^t, b_{g,2}^t, \cdots, b_{g,m}^t, \cdots, b_{g,M}^t\}$ represents the current optimal solution that can be found in the current search algorithm. According to (19) , the nest position is reserved for the next generation.

Because the location information of the solution space is a continuous value, it cannot be directly used by the optimization objective function to solve the optimal value. Therefore, the effect function should be constructed to map the continuous value of position to the binary discrete value between $\{0, 1\}$. The expression of the effect function H is as follows:

$$H(x_{i,m}(t+1)) = \frac{x_{i,m}^{t+1} - x_m^{min}}{x_m^{max} - x_m^{min}}, \tag{20}$$

$$x_{i,m}^{t+1} = \begin{cases} 1, & H(x_{i,m}(t+1)) < \gamma \\ 0, & \text{otherwise}, \end{cases} \tag{21}$$

where x_m^{min} is the minimum value of m dimension in the nest position, and x_m^{max} is the maximum value of m dimension in the nest position. Furthermore, γ is uniformly distributed, and $\gamma \sim U(0, 1)$. When the host bird finds the cuckoo's egg (probability P_a), it abandons the nest. Therefore, after the complete local search by Levy flight, some solutions should be searched again to update their positions and retain a superior set of solutions. In the process of local search, to obtain the difference between the current individual and the excellent individual in the population, $x_{i,m}^{t}$ is carried out differential evolution. The individual obtained after evolution has more genetic information, rendering it close to the excellent individual in the population. The specific process is as follows:

1) Mutation Operation: The mutation operation is used to obtain individual variation through the difference strategy to obtain the genetic information of multiple individuals. The expression of individual variation $u_{i,m}^{t}$ is as follows:

$$u_{i,m}^{t} = x_{i,m}^{t} + \kappa \cdot (x_{p,m}^{t} - x_{q,m}^{t}),\ p, q = 1, 2, \cdots, K+N, \tag{22}$$

where κ is the scaling factor, $x_{p,m}^{t}$ represents the value of the p^{th} nest in the m dimension of the nest position in the t generation, and $x_{q,m}^{t}$ represents the value of the q^{th} nest in the m dimension of the nest position in the t generation.

C. Crossover Operation

The crossover operation generates candidate individual $v_{i,m}^{t}$ by crossing between the parent and the mutant, which ensures that at least one set of information in the next generation of individual information originates from the mutant. The expression of candidate individual $v_{i,m}^{t}$ is as follows:

$$v_{i,m}^{t} = \begin{cases} u_{i,m}^{t}, & \alpha_4 < CR \text{ or } m = \beta_4 \\ x_{i,m}^{t}, & \text{otherwise}, \end{cases} \tag{23}$$

where $CR \in [0, 1]$ is the crossover probability, $\alpha_4 = rand(0, 1)$ is the random number generated during $[0, 1]$, $\beta_4 = unidrnd(M)$ represents a random positive integer generated during $[1, M]$.

D. Select Operation

After the population mutation and crossover operation, the position is discretized, and the dominant relationship between the individual $v_{i,m}^{t}$ and $x_{i,m}^{t}$ the parent is determined by comparing the size of the optimization objective function . Therefore, a new generation of individuals $x_{i,m}^{t+1}$ to inherit the dominant individuals to the next generation:

$$x_{i,m}^{t+1} = \begin{cases} v_{i,m}^{t}, & T(v_{i,m}^{t}) < T(x_{i,m}^{t}) \\ x_{i,m}^{t}, & \text{otherwise}. \end{cases} \tag{24}$$

The specific flow of the discrete cuckoo search algorithm based on differential evolution is stated in Algorithm 2.

Algorithm 2 Discrete cuckoo search algorithm based on differential evolution

Input: system parameters, including MV, CV, AP, service cache matching results Φ, computing tasks $T_m = \{\omega_m, t_m, s_m^{in}, s_m^{out}, SR_{fm}, c_m\}$, $r_{m,n}$ and other indicators. cuckoo algorithm parameters, including nest location set $\{x_1, x_2, \cdots, x_{K+N}\}$, t_{max}, P_a and other indicators.
Initialization: Initialize the nest position and other parameters, and record the current optimal solution.
Begin:
for $t < t_{max}$ **do**:
Calculate and update all nest positions $x_{i,m}^{t+1}$ according to (18);
Binary discretization of all nest locations $x_{i,m}^{t+1} \sim (0, 1)$;
According to (15), the optimization objective function value T is calculated;
if T is the optimal value in the current iteration **then**:
The solution space associated with is the optimal solution space and retaining the current nest position;
else
The current nest position is not retained;
end if
if $rand(0, 1) > P_a$ **then**:
Perform mutation, crossover and selection operations to locally update the nest position;
Binary discretization of all nest locations $x_{i,m}^{t+1} \sim (0, 1)$;
According to (15), the optimization objective function value T' is calculated;
if T' is the optimal value in the current iteration **then**:
The solution space associated with T' is the optimal solution space and retaining the current nest position;
else
The current nest position is not retained;
end if
end if
end for
The current optimal solution space is the task uninstallation strategy;
Output the optimal task uninstallation strategy x^*.
End

V. Performance Analysis

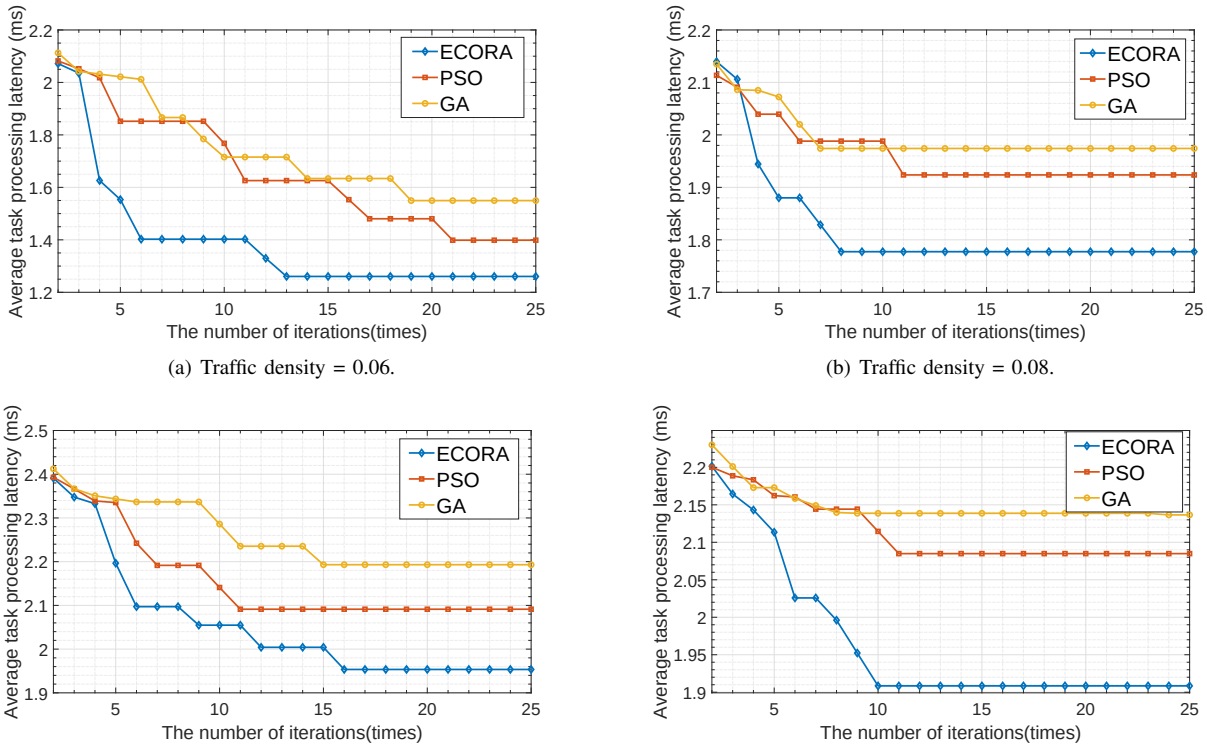
Simulations were performed using MATLAB for the two-way six-lane traffic scenario to compare the convergence properties of the GA strategy and PSO strategy. The influence of various traffic conditions on the quality of service, task processing delay, and other indicators were investigated. The influence of two key parameters (P_α and α_3) on the convergence of the proposed algorithm is analyzed, and a series of simulations performed. Furthermore, the input data range of the task generated by the mission vehicle is $[0.3, 0.45]$ Mb/task, and the calculation demand range of the task is $[0.3, 0.45]$ GHz/task. The specific simulation parameters are summarized in Table. I.

TABLE I
Simulation parameters.

Symbol	Parameters	Value
$P_{m,k}^t$	AP transmitting power and antenna gain	16dB
$P_{m,n}^t$	Vehicle transmitting power and antenna gain	3dB
W	Total channel bandwidth	20MHZ
θ_{th}	Threshold of the received signal power	5dB
σ	Standard deviation of shadow fading	5
F	Number of cache service types	4
ρ_{th}^f	Idle rate threshold	0.1
N	Number of the collaborative vehicles	18
M	Number of the mission vehicles	45
r	Effective communication radius of vehicle	100m
R	Effective communication radius of AP	150m

The simulation compares the variation rule of average task processing delay in various resource allocation strategies with different traffic density, as displayed in Fig. 2. The horizontal axis represents the number of iterations, and the vertical axis represents the average task processing delay. Furthermore, the ECORA, PSO, and GA strategy are represented by rhombus, square, and circle curves in the figure Fig. 2(a)∼Fig. 2(d) reveals that the average task processing delay of the ECORA strategy, PSO strategy, and GA strategy increases with the increase in the traffic density, and the average task processing delay of ECORA strategy is the lowest because with the increase in the traffic density, the resources of the system are limited. When the number of mission vehicles with computing requirements increases, the computing resources available to each mission vehicle decreases, and subsequently, the delay of task processing increases. Furthermore, the PSO strategy and GA strategy ignore the uneven traffic density and the effect of increasing traffic density on the resource allocation process, which results in an unbalanced AP load. When the traffic density increases, the number of mission vehicles near some AP increases sharply, but the cooperative vehicles around the AP do not consider the service caching strategy, and idle computing resources cannot be utilized. In this case, the limited computing resources of the AP cannot satisfy the unloading requirements of surrounding mission vehicles, which results in high task processing delay. The ECORA strategy can provide popular cache service for collaborative vehicles in advance according to the position of vehicles, driving direction, and other mobile attributes, combined with social contact between vehicles. Ensure that cooperative vehicles exist around the mission vehicle that can unload the specified task to improve the success rate of mission unloading of the mission vehicle. Next, the influence of traffic density on task processing delay is effectively reduced, and the task processing delay is low. Furthermore, Fig. 2(b) reveals that when traffic density is 0.08 vehicles/m, the average delay of task processing of ECORA strategy is 1.777 ms, whereas those of the PSO and GA strategies is 1.923 ms and 1.974 ms, respectively. The overall task processing delay of the algorithm improved by 7.59% and 9.98%, respectively.

The simulation compares the variation rules of average task processing delay in various resource allocation strategies at different average vehicle speeds, as displayed in Fig. 3. In the figure, the horizontal axis represents the number of iterations, and the vertical axis represents the average task processing delay. Furthermore, addition, ECORA, PSO, and GA strategies are represented by rhombus, square, and circle curves in the figure. As displayed in Fig. 3(a)∼Fig. 3(d), the average task processing delay of the ECORA, PSO, and GA strategies increases slowly with the increase in the average vehicle speed, and the average task processing delay of ECORA strategy is the lowest because with the increase in the average speed, the duration of communication between vehicles and between



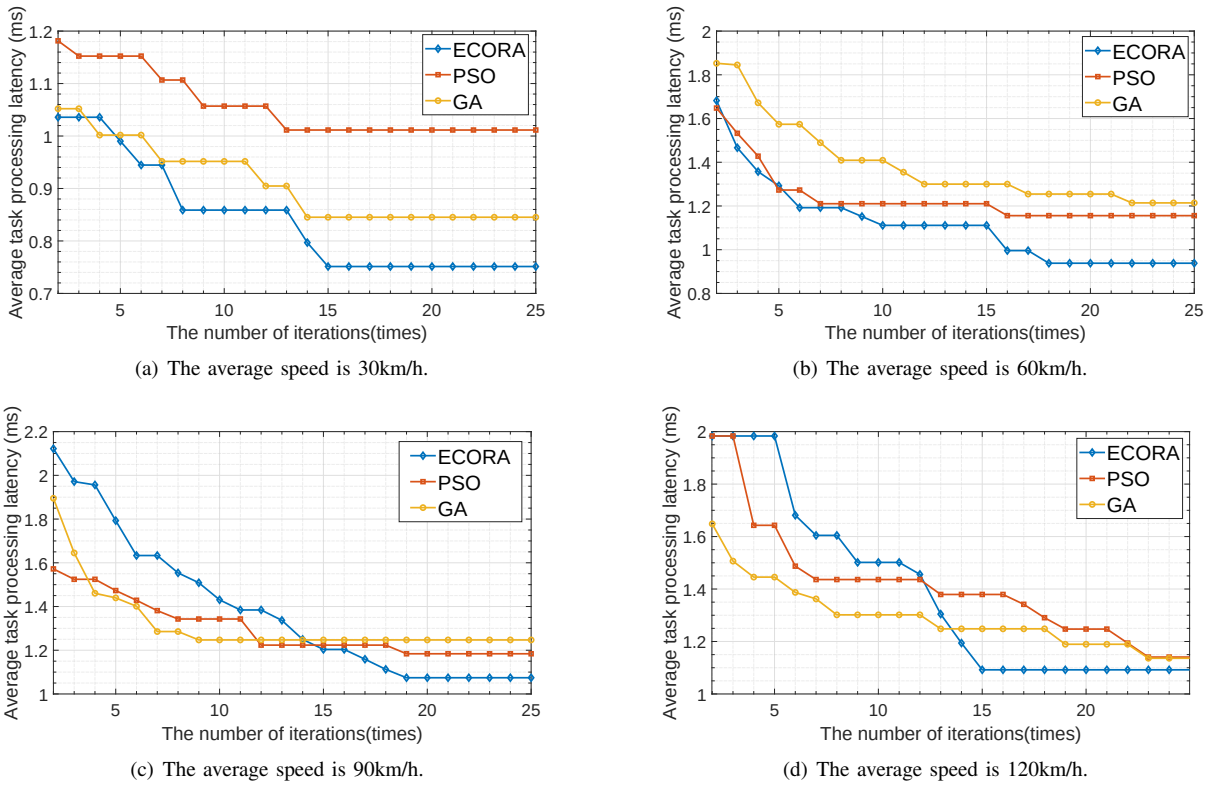
(a) Traffic density = 0.06.

(b) Traffic density = 0.08.

(c) Traffic density = 0.10.

(d) Traffic density = 0.12.

Fig. 2. Change of the average task processing delay under various traffic density conditions, here ECORA is our proposed scheme, PSO is from [23], and GA is from [24].



(a) The average speed is 30km/h.

(b) The average speed is 60km/h.

(c) The average speed is 90km/h.

(d) The average speed is 120km/h.

Fig. 3. Average task processing delay under various average speed, here ECORA is our proposed scheme, PSO is from [23], and GA is from [24].

vehicles and AP decreases, which leads to an increase in the probability of communication link interruption and the failure rate of task unloading, which increases the delay of task processing. Furthermore, the PSO and GA strategies ignore

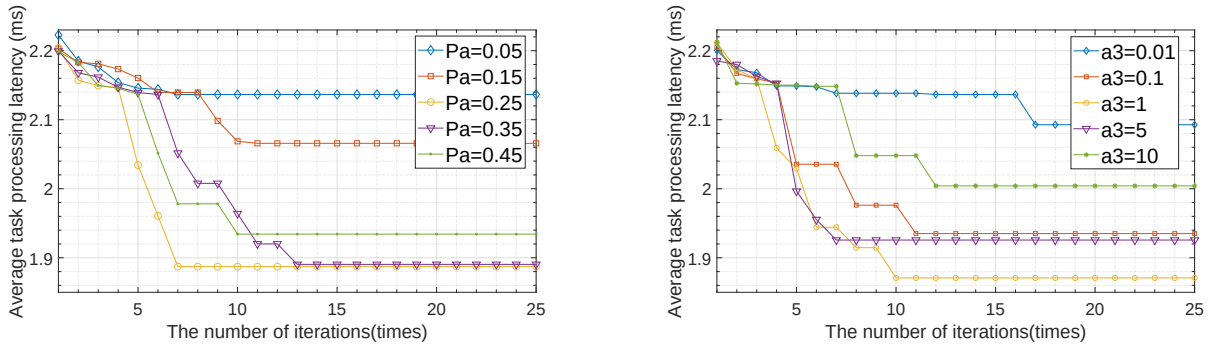
the influence of social connection and mobile attribute on resource allocation process. When the average vehicle speed increased, the duration of the communication link between mission vehicle and cooperative vehicle decreased, and fewer cooperative vehicles were present around mission vehicle to cache and unload mission service. Thus, more mission vehicles unload tasks to AP, which results in an increased AP load and higher task processing time. In ECORA strategy, cooperative vehicles follow the service caching principle of high mobility similarity and high social similarity. Cooperative vehicles with sufficient caching services around the mission vehicles provide computing resources. The discrete cuckoo search algorithm based on differential evolution was used to allocate resources efficiently and determine the unloading position of the task. Effectively reducing the effect of high-speed vehicle movement on task processing delay, and the task processing delay is low. Furthermore, Fig. 3(b) reveals that when the average speed is 60 km/h, the average delay of task processing of ECORA strategy is 0.938 ms, whereas those of PSO and GA strategies is 1.156 ms and 1.214 ms, respectively. Thus, the overall task processing delay of the algorithm improved by 18.86% and 22.73% respectively.

The variation rule of average task processing delay in ECORA algorithm is shown in Fig.4 when different algorithm parameters are simulated and compared. In the figure, the horizontal axis represents the number of iterations of the algorithm, and the vertical axis represents the average task processing delay. In addition, in Fig.4(a), curves with rhombus, square, circle, triangle and dot respectively represent the change of average task processing delay when P_α is 0.05, 0.15, 0.25, 0.35 and 0.45. Similarly, in Fig.4(b), curves with rhombus, square, circle, triangle and dot represent the change of average task processing delay when the step factor α_3 of ECORA strategy is 0.01, 0.1, 1, 5 and 10 respectively. As can be seen from Fig.4(a), as the number of iterations increases, when the P_α value is 0.05, the convergence speed is the slowest and the average task processing delay is the largest. This is because when the P_α value is too low, the update frequency of local search decreases and the update frequency of the optimal solution of unloading task decreases. In this case, the convergence speed is the slowest, and the algorithm is easy to fall into local optimal, which leads to the maximum average task processing delay. In addition, it can be seen from Fig.4(b) that with the increase of iteration times, the algorithm has the best convergence capability when the value α_3 is 1. At this point, the convergence speed is the highest and the average task processing delay is the lowest. This is because when the α_3 value is too large, different bird's nest positions (equation solutions) are far away from each other after the global search in the initial iteration, and it is difficult to determine the optimal solution by moving the bird's nest positions in a small range in the subsequent local search. As a result, the probability of finding the optimal solution decreases, the convergence speed is slow, and obtaining the optimal solution using the algorithm is difficult. Similarly, if the α_3 value is too small, different bird's nest locations will be close to each other after the global search in the initial iteration, and it is difficult to determine the optimal solution by moving the bird's nest location in a small range in the subsequent local search. As a result, the probability of finding the optimal solution decreases, the convergence speed is slow, and using the algorithm to obtain the optimal solution is difficult.

The load state distribution of all APS in the same section of road at the same time with different strategies were simulated. The results are displayed in Fig.5. In the figure, the horizontal axis represents the ID of the AP and the vertical axis represents the load status of the AP. Furthermore, from the top to bottom in the bar chart are the GA, PSO, and ECORA strategies. The load value of the GA and PSO strategies at AP_3 is considerably higher than 10, and the load value at $\{AP_1, AP_2\}$ is 0. However, the load value of ECORA policy is not 0 at any AP and is less than 4 at any AP because in the GA and PSO strategies, communication link reliability and processing delay are considered as the basis of the optimal solution of task unloading for mission vehicles, ignoring the influence of service caching process of cooperative vehicles on the success rate of task unloading. In fact, PSO and GA strategies cannot allocate idle caching resources for cooperative vehicles based on mobile social connections. When the number of cooperative vehicles is reduced, the success rate of task unloading between mission vehicles and cooperative vehicles drops sharply, thus reducing the computational resource utilization rate of cooperative vehicles. Therefore, considering the uneven distribution of traffic flow density, there are more cooperative vehicles near $\{AP_1, AP_2\}$ and less cooperative vehicles near AP_3. Meanwhile, PSO and GA strategies cannot make full use of computing resource utilization of cooperative vehicles around AP_3, resulting in waste of the computing resources of cooperative vehicles and increasing the load of AP_3. In ECORA, the number of cooperative vehicles near $\{AP_1, AP_2\}$ is large, and the mission vehicles there can unload tasks to select cooperative vehicles. Thus, the $\{AP_1, AP_2\}$ load is small. The ECORA strategy detects popular task unloading types and performs service caching through social contact and mobile contact between vehicles to ensure that many cooperative vehicles around mission vehicles that can perform unloading services. According to the constraints of load limit and the maximum task completion time and others, the optimal solution of unloading scheme is calculated. Under the condition of load balancing, the average delay of vehicle task processing is reduced and the quality of communication service is improved. Therefore, the load state distribution of AP under this strategy is more reasonable. Therefore, the ECORA algorithm can effectively balance AP load.

VI. Conclusion and Future Work

In this study, we proposed an efficient caching and offloading resource allocation strategy in vehicular social networks. First, we studied the influence of social contact and motion state between vehicles on the service cache and task unloading process. Next, we established the social vehicular network communication system model of joint service caching and task unloading, thus optimizing the utilization rate of storage and computing resources under the cooperation mechanism



(a) Influence of parameters P_α on the strategy.

(b) Influence of parameters α_3 on the strategy.

Fig. 4. Average task processing delay changes under different strategy parameters.

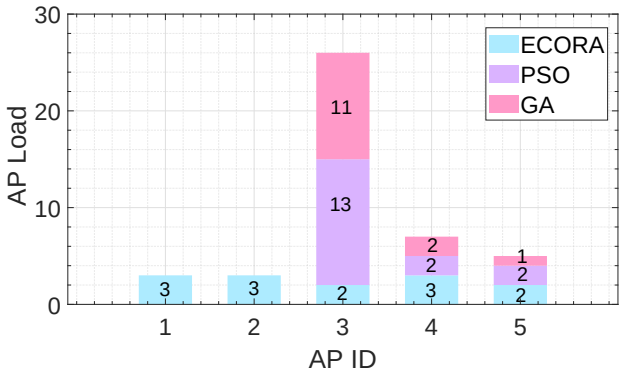


Fig. 5. RSU load distribution, here ECORA is our proposed scheme, PSO is from [23], and GA is from [24].

between vehicles and improving service quality. To minimize the average task processing delay for achieving high quality of service, we studied the combined resource allocation optimization problem of joint service caching and task unloading, and categorized the NP-hard problem into two sub-problems to be solved separately. A stable matching algorithm based on mobile social connections and a discrete cuckoo search algorithm based on differential evolution were designed. By dynamically updating the cache resource allocation scheme, the optimal task unloading scheme was adaptively selected to maximize the quality of service and minimize the average task processing delay. The simulation results revealed that compared with PSO and GA strategies, the ECORA strategy can effectively reduce the average task processing delay, improve the quality of user data communication service, and effectively achieve RSU load balancing. In the future, we will develop an accurate resource allocation model for the social vehicular network so that the 5G social network can provide superior services for time-sensitive and computation-intensive intelligent transportation scenarios such as unmanned driving and remote driving.

REFERENCES

[1] S. Kudva, S. Badsha, S. Sengupta, I. Khalil, and A. Zomaya, "Towards secure and practical consensus for blockchain based vanet," *Information Sciences*, vol. 545, pp. 170–187, 2021.

[2] T. Zhang, "Toward automated vehicle teleoperation: Vision, opportunities, and challenges," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11 347–11 354, 2020.

[3] A. Kousaridas, R. P. Manjunath, J. Perdomo, C. Zhou, E. Zielinski, S. Schmitz, and A. Pfadler, "Qos prediction for 5g connected and automated driving," *IEEE Communications Magazine*, vol. 59, no. 9, pp. 58–64, 2021.

[4] T. do Vale Saraiva, C. A. V. Campos, R. dos Reis Fontes, C. E. Rothenberg, S. Sorour, and S. Valaee, "An application-driven framework for intelligent transportation systems using 5g network slicing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 5247–5260, 2021.

[5] M. Liu, T. Song, J. Hu, J. Yang, and G. Gui, "Deep learning-inspired message passing algorithm for efficient resource allocation in cognitive radio networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 641–653, 2018.

[6] F. Tang, B. Mao, N. Kato, and G. Gui, "Comprehensive survey on machine learning in vehicular network: technology, applications and challenges," *IEEE Communications Surveys & Tutorials*, 2021.

[7] M. D. Hossain, T. Sultana, W. ur Rahman, G.-W. Lee, and E.-N. Huh, "Game theory based dynamic computation offloading in mec-enabled vehicular networks," *KIISE Transactions on Computing Practices*, vol. 28, no. 4, pp. 216–224, 2022.

[8] Z. Tong, X. Deng, F. Ye, S. Basodi, X. Xiao, and Y. Pan, "Adaptive computation offloading and resource allocation strategy in a mobile edge computing environment," *Information Sciences*, vol. 537, pp. 116–131, 2020.

[9] E. Karimi, Y. Chen, and B. Akbari, "Task offloading in vehicular edge computing networks via deep reinforcement learning," *Computer Communications*, vol. 189, pp. 193–204, 2022.

[10] G. Wang, F. Xu, H. Zhang, and C. Zhao, "Joint resource management for mobility supported federated learning in internet of vehicles," *Future Generation Computer Systems*, vol. 129, pp. 199–211, 2022.

[11] Y. Zhang, Y. Zhao, and Y. Zhou, "User-centered cooperative communication strategy for 5g internet of vehicles," *IEEE Internet of Things Journal*, 2022.

[12] Y. Wang, P. Lang, D. Tian, J. Zhou, X. Duan, Y. Cao, and D. Zhao, "A game-based computation offloading method in vehicular multiaccess edge computing networks," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4987–4996, 2020.

[13] Z. Xiao, X. Dai, H. Jiang, D. Wang, H. Chen, L. Yang, and F. Zeng, "Vehicular task offloading via heat-aware mec cooperation using game-theoretic method," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2038–2052, 2019.

[14] P. Dai, K. Hu, X. Wu, H. Xing, F. Teng, and Z. Yu, "A probabilistic approach for cooperative computation offloading in mec-assisted vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[15] X. Xiao, X. Hou, X. Chen, C. Liu, and Y. Li, "Quantitative analysis for capabilities of vehicular fog computing," *Information Sciences*, vol. 501, pp. 742–760, 2019.

[16] Y. Yin, M. Liu, G. Gui, H. Gacanin, H. Sari, and F. Adachi, "Cross-layer resource allocation for uav-assisted wireless caching networks with noma," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3428–3438, 2021.

[17] F. Zhao, Y. Chen, Y. Zhang, Z. Liu, and X. Chen, "Dynamic offloading and resource scheduling for mobile-edge computing with energy harvesting devices," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 2154–2165, 2021.

[18] J. Feng, Q. Pei, F. R. Yu, X. Chu, J. Du, and L. Zhu, "Dynamic network slicing and resource allocation in mobile edge computing systems," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7863–7878, 2020.

[19] X. Huang, K. Xu, C. Lai, Q. Chen, and J. Zhang, "Energy-efficient offloading decision-making for mobile edge computing in vehicular networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2020, no. 1, pp. 1–16, 2020.

[20] Z. Ji, X. Lu, R. Yin, and C. Wu, "Unlicensed assisted transmission in vehicular edge computing networks," *EURASIP Journal on Advances in Signal Processing*, vol. 2022, no. 1, pp. 1–20, 2022.

[21] X.-Q. Pham, T.-D. Nguyen, V. Nguyen, and E.-N. Huh, "Joint node selection and resource allocation for task offloading in scalable vehicle-assisted multi-access edge computing," *Symmetry*, vol. 11, no. 1, p. 58, 2019.

[22] F. Wang and M. Xie, "Global intelligent and connected vehicles forecast report (2020-2024)." IDC, Tech. Rep., 2018.

[23] X. Hou, Z. Ren, J. Wang, W. Cheng, Y. Ren, K.-C. Chen, and H. Zhang, "Reliable computation offloading for edge-computing-enabled software-defined iov," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7097–7111, 2020.

[24] H. Li, H. Xu, C. Zhou, X. Lü, and Z. Han, "Joint optimization strategy of computation offloading and resource allocation in multi-access edge computing environment," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 10 214–10 226, 2020.

[25] G.-F. Hong, W. Su, Q.-L. Wen, and P.-L. Wu, "Ravec: An optimal resource allocation mechanism in vehicular mec systems." *Journal of Information Science & Engineering*, vol. 36, no. 4, 2020.

[26] Z. Wang and R. Hou, "Joint caching and computing resource allocation for task offloading in vehicular networks," *IET Communications*, vol. 14, no. 21, pp. 3820–3827, 2020.

[27] Y. Zhang, M. Zhang, C. Fan, F. Li, and B. Li, "Computing resource allocation scheme of iov using deep reinforcement learning in edge computing environment," *EURASIP Journal on Advances in Signal Processing*, vol. 2021, no. 1, pp. 1–19, 2021.

[28] H. Huang, M. Liu, G. Gui, H. Gacanin, H. Sari, and F. Adachi, "Unsupervised learning-inspired power control methods for energy-efficient wireless networks over fading channels," *IEEE Transactions on Wireless Communications*, vol. 21, no. 11, pp. 9892–9905, 2022.

[29] Y. Zhu, Y. Hu, and A. Schmeink, "Delay minimization offloading for interdependent tasks in energy-aware cooperative mec networks," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2019, pp. 1–6.

[30] H. Guo, J. Liu, and J. Zhang, "Efficient computation offloading for multi-access edge computing in 5g hetnets," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[31] M. Mao, T. Hu, and W. Zhao, "Reliable task offloading mechanism based on trusted roadside unit service for internet of vehicles," *Ad Hoc Networks*, vol. 139, p. 103045, 2023.

[32] G. Wang, F. Xu, and C. Zhao, "Qos-enabled resource allocation algorithm in internet of vehicles with mobile edge computing," *IET Communications*, vol. 14, no. 14, pp. 2326–2333, 2020.

[33] Y. Yin, M. Liu, G. Gui, H. Gacanin, and H. Sari, "Minimizing delay for mimo-noma resource allocation in uav-assisted caching networks," *IEEE Transactions on Vehicular Technology*, 2022.

[34] S. Song, C. Lee, H. Cho, G. Lim, and J.-M. Chung, "Clustered virtualized network functions resource allocation based on context-aware grouping in 5g edge networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 5, pp. 1072–1083, 2019.

[35] F. Hussain, R. Hussain, A. Anpalagan, and A. Benslimane, "A new block-based reinforcement learning approach for distributed resource allocation in clustered iot networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 2891–2904, 2020.

[36] Y. Qian, L. Hu, J. Chen, X. Guan, M. M. Hassan, and A. Alelaiwi, "Privacy-aware service placement for mobile edge computing via federated learning," *Information Sciences*, vol. 505, pp. 562–570, 2019.

[37] S. Pradhan and S. Tripathy, "Frac: a flexible resource allocation for vehicular cloud system," *IET Intelligent Transport Systems*, vol. 14, no. 14, pp. 2141–2150, 2020.

[38] T. Kim, H. Min, E. Choi, and J. Jung, "Optimal job partitioning and allocation for vehicular cloud computing," *Future Generation Computer Systems*, vol. 108, pp. 82–96, 2020.

[39] X. Gu, W. Duan, G. Zhang, Y. Ji, M. Wen, and P.-H. Ho, "Socially aware v2x networks with ris: Joint resource optimization," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 6, pp. 6732–6737, 2022.

[40] X. Huang, L. He, X. Chen, L. Wang, and F. Li, "Revenue and energy efficiency-driven delay constrained computing task offloading and resource allocation in a vehicular edge computing network: a deep reinforcement learning approach," *IEEE Internet of Things Journal*, 2021.

[41] J. Feng, Z. Liu, C. Wu, and Y. Ji, "Mobile edge computing for the internet of vehicles: Offloading framework and job scheduling," *IEEE vehicular technology magazine*, vol. 14, no. 1, pp. 28–36, 2018.

[42] J. Zhao, P. Dong, X. Ma, X. Sun, and D. Zou, "Mobile-aware and relay-assisted partial offloading scheme based on parked vehicles in b5g vehicular networks," *Physical Communication*, vol. 42, p. 101163, 2020.

[43] L. Xu, Z. Yang, H. Wu, Y. Zhang, Y. Wang, L. Wang, and Z. Han, "Socially driven joint optimization of communication, caching, and computing resources in vehicular networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 1, pp. 461–476, 2021.

[44] H. Liang, X. Zhang, X. Hong, Z. Zhang, M. Li, G. Hu, and F. Hou, "Reinforcement learning enabled dynamic resource allocation in the internet of vehicles," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4957–4967, 2020.