Energy-Delay Minimization of Task Migration Based on Game Theory in MEC-Assisted Vehicular Networks

Haipeng Wang, *Student Member, IEEE*, Tiejun Lv [ID], *Senior Member, IEEE*, Zhipeng Lin [ID], *Member, IEEE*, and Jie Zeng [ID], *Senior Member, IEEE*

Abstract—Roadside units (RSUs), which have strong computing capability and are close to vehicle nodes, have been widely used to process delay- and computation-intensive tasks of vehicle nodes. However, due to their high mobility, vehicles may drive out of the coverage of RSUs before receiving the task processing results. In this paper, we propose a mobile edge computing-assisted vehicular network, where vehicles can offload their tasks to a nearby vehicle via a vehicle-to-vehicle (V2V) link or a nearby RSU via a vehicle-to-infrastructure link. These tasks are also migrated by a V2V link or an infrastructure-to-infrastructure (I2I) link to avoid the scenario where the vehicles cannot receive the processed task from the RSUs. Considering mutual interference from the same link of offloading tasks and migrating tasks, we construct a vehicle offloading decision-based game to minimize the computation overhead. We prove that the game can always achieve Nash equilibrium and convergence by exploiting the finite improvement property. We then propose a task migration (TM) algorithm that includes three task-processing methods and two task-migration methods. Based on the TM algorithm, computation overhead minimization offloading (COMO) algorithm is presented. Extensive simulation results show that the proposed TM and COMO algorithms reduce the computation overhead and increase the success rate of task processing.

Index Terms—Computation offloading, mobile edge computing, game theory, task migration, I2I.

I. INTRODUCTION

THE Internet of Vehicles (IoV), which originated from the Internet of Things (IoT), enables vehicle-to-everything (V2X) communication to enhance intelligent driving services, thereby improving the level of social traffic service intelligence [1], [2] and the management of urban flows [3]. However, due to their limited size, vehicles generally do not offer sufficient computing resources for their service requirements with low computing latency [4]. As a result, efficient networks must be designed to address computing resource shortages on the vehicle side.

Considering the rich computing resources provided by the Internet, cloud-based in-car networks have been proposed to address the explosive growth of computing task requirements of vehicles. The cloud network uses advanced communication technologies to integrate various resources and provides assistance for task offloading. Vehicles can either process computing tasks locally or offload tasks to the cloud. The cloud network improves resource utilization and quality of service (QoS) [5], [6]. However, with the increase in the number of smart devices, the number of computing tasks has also increased. These devices also need to offload computing tasks to the cloud for processing, which inevitably causes network congestion and resource waste.

Mobile edge computing (MEC) is considered the most promising computing paradigm to improve the QoS of users. MEC effectively integrates wireless network and Internet technologies, adds computing, storage, processing and other functions on the wireless network side, and uses its geographical advantages to provide more convenient services for vehicles. MEC can improve network utilization efficiency and achieve low latency and energy consumption. In MEC networks, vehicles can offload tasks to nearby vehicles with idle resources or to nearby roadside units (RSUs) for processing [7]. However, when the vehicle offloads computing tasks to the RSU, due to the high mobility of the vehicle and the increasing size of the computing tasks, the vehicle cannot receive the processing results of computing tasks from the RSU because the vehicle has already driven out of the RSU's communication coverage.

To receive the processing results from the RSU, the computing tasks must be migrated from vehicles. Typically, there are two ways for vehicles to offload their tasks to an RSU with an MEC server: vehicle-to-infrastructure (V2I) mode and vehicle-to-vehicle (V2V)+V2I mode. Correspondingly, the RSU also has two ways to migrate the computing results, i.e., infrastructure-to-infrastructure (I2I)+infrastructure-to-vehicle (I2V) mode and I2V+V2V mode. Because the size of each offloaded computing

task is different, the computing tasks require different computing and transmission resources. Using the same channel to carry out the offloading tasks or transmitting the computing results will cause common channel interference [8]–[10], which will reduce the transmission/migration rate and increase energy consumption and delay. Therefore, the energy consumption and delay must be balanced, and the tasks must be completed within a limited time.

In this paper, we propose an MEC-assisted vehicular network architecture. We optimize the task offloading and migration decisions in the vehicle network to minimize the weighted sum of delay and energy consumption. In the proposed system, we propose a task migration (TM) algorithm and a computation overhead minimization offloading (COMO) algorithm based on game theory. The key contributions of this paper are summarized as follows:

- To reduce the computation overhead, we construct an MEC-assisted vehicular network architecture, where the co-channel interference of task offloading and task migration is considered.
- Based on game theory, we formulate the constrained optimization problem of offloading decisions as a game. We prove that the game can always achieve Nash equilibrium (NE) and can converge according to the finite improvement property (FIP).
- We propose a TM algorithm and a COMO algorithm that can reduce the computation overhead and increase the success rate of task processing.

The rest of this paper is organized as follows: Section II reviews related work, followed by the MEC-assisted vehicular network system model in Section III. In Section IV, we construct a game-based vehicle offloading decision and propose a TM algorithm and a COMO algorithm. Section V discusses the simulation results. Finally, we summarize our paper in Section VI.

II. RELATED WORK

In recent years, MEC-based task offloading and task migration have drawn the attention of researchers. The latency [11]–[14] and the energy consumption of the system [15]–[18] are two criteria used to evaluate the performance of task offloading and task migration.

A. Cloud-Based Computation Offloading

Since task offloading can effectively alleviate the burden of insufficient vehicle computing resources, cloud-based computation offloading has been applied to various situations. In vehicular fog and cloud computing (VFCC) systems, to address the impact of vehicles leaving unfinished tasks, the authors in [19] described the task offloading problem as a semi-Markov decision process (SMDP), and the value iteration algorithm of the SMDP was designed to maximize the total long-term benefits of the VFCC system. To effectively improve the offloading efficiency, a method of combining resource allocation and offloading was proposed in [20], where a low-complexity algorithm was developed to jointly optimize the offloading decision and resource allocation. Wang *et al.* [21] formulated the offloading problem

as an optimization problem and proposed a fog-cloud computational offloading algorithm and a heuristic algorithm to jointly minimize the power consumption of vehicles and that of the computational facilities.

B. Fog-Based Computation Offloading

Fog computing, as a transition between cloud computing and edge computing, concentrates data, processing, and applications in devices at the edge of the network, rather than storing them in the cloud. To detect and take necessary steps for public safety during a disaster, the authors in [22] proposed crowdsourcing-based disaster management using a fog computing model in the IoT. Zhang *et al.* [23] proposed a regional cooperative fog-computing-based intelligent vehicular network architecture and a hierarchical model with intra-fog and inter-fog resource management to optimize the energy efficiency and packet dropping rates. To minimize the average offloading delay, the task offloading problem of dynamic fog networks was strictly defined as an online stochastic optimization problem. In [24], the authors designed task offloading strategies for stationary status and nonstationary status algorithms and proposed the use of a discount factor to reduce the delay.

C. MEC-Based Computation Offloading

In the MEC system, vehicles can offload computation tasks to other devices that have strong computing power and resources to reduce the system energy consumption and delay [25]. Considering the mutual interference of tasks in the same channel, the authors in [26] and [27] constructed MEC-assisted networks and proposed algorithms based on game theory to minimize the network computation overhead. To meet the requirements of mobility and energy harvesting of an IoT network, the authors in [28] proposed an online mobility-aware offloading and a resource allocation algorithm that can balance the system service cost and energy queue length. An efficient algorithm based on submodular optimization and a collaborative task computing scheme were proposed in [29] to efficiently mitigate data redundancy, conserve network bandwidth consumption and reduce the cost of processing tasks.

D. Task Migration

With the development of science and technology, the speed of vehicles has substantially improved, and the size of the data that needs to be processed has increased. To effectively address the resulting tasks from vehicle offloading, efficient task migration strategies must be designed such that vehicles can offload their tasks to the RSU for processing, where the RSU determines the task processing mode by combining the delay threshold and energy consumption requirements of the task.

1) Task Migration in Traditional Optimization Mode: In [30], to achieve an optimal balance between energy consumption and time cost during service migration, the authors designed a 6th generation mobile network (6 G)-enabled box task migration method and a strength Pareto evolutionary algorithm for the IoV. The authors in [31] establish an efficient

service migration model and build a nonlinear 0-1 programming problem by designing a business migration scheme based on a particle swarm, effectively reducing the latency and energy consumption. In [32], by introducing distributed traffic guidance in large MEC systems and distinguishing between two different types of network elements, the scalability problem of a large MEC network was resolved as a partitioned MEC network, namely, edge servers and routers. Then, dynamic shortest path selection and dynamic multipath searching algorithms were proposed to achieve high QoS with ultra-low latency.

2) Task Migration in Deep Learning Mode: In [33], the authors propose a multi-agent deep reinforcement learning algorithm that maximizes the comprehensive utility of communication, computing and routing planning in a distributed manner, thus reducing service delay, migration cost and travel time. The authors in [34] proposed a deep Q-learning service migration decision algorithm and a neural network-based service migration framework that realizes the adaptive migration of task offloading when connected vehicles move. To minimize the data processing cost within the system and ensure the delay constraints of applications, the author in [35] formulates a unified communication, computing, caching and collaborative computing framework and develops a cooperative data scheduling scheme to model the data scheduling as a deep reinforcement learning problem which is solved by an enhanced deep Q-Network algorithm with a single target Q-network.

Different from these aforementioned studies, our work considers the mutual interference of tasks in the same channel, whenever the channel processes the common channel offloading tasks or the common channel transmitting computing results. On this basis, a TM algorithm and a COMO algorithm based on game theory are proposed to reduce the computation overhead and increase the success rate of task processing according to the offloading decisions of the vehicle.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we first introduce the MEC-assisted vehicular network. Then, we analyze the weighted sum of delay and energy consumption when the vehicles are offloading and migrating tasks.

A. System Model

As shown in Fig. 1, we consider an MEC-assisted vehicular network system with multiple vehicles and S RSUs, where the set of RSUs is $\mathbf{RSU} = \{RSU_1, RSU_2, \ldots, RSU_S\}$. In the proposed network, we assume that vehicles are equipped with 802.11p interfaces and network interfaces. Every vehicle is equipped with on-board units (OBUs) [36], GPS, wireless communication devices and other devices. GPS can provide real-time information about the current positions and directions of vehicles [37]. The OBU is a microwave device that uses dedicated short-range communication technology to communicate with the RSU: the OBU has limited computing power and storage capacity to process computing tasks with small data size. The RSUs, which have strong computing power, are used as edge servers.
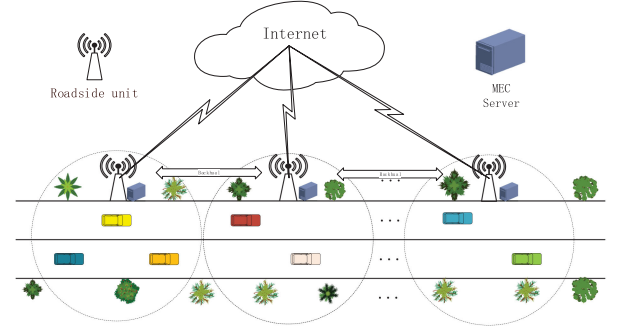

Fig. 1. An MEC-assisted vehicular network system.

In the MEC-assisted vehicular network, we use software defined networks (SDN) technology. An SDN, which has deep programmability, has a physical data plane and an abstracted control plane [38]. In the network, the sub-SDN controller collects idle vehicle information and target vehicle computing task information. Idle vehicle information includes vehicle location, driving direction, and CPU frequency of the vehicle server. The target vehicle calculation task information includes task size, calculation workload and maximum delay tolerance. The vehicle usually periodically sends the information to the nearest RSU. The RSUs store the collected information in the data center for invocation by the SDN controller [39].

According to the unit circle protocol model [40], the communication range of each RSU is a circle with radius $\frac{L_{I2I}}{2}$. In the system, we use a single-input and single-output orthogonal frequency division multiple access scheme to avoid unnecessary interference between RSUs. The set of vehicles is $\mathbf{M} = \{1, 2, \ldots, M\}$, and we use m to denote a vehicle that has a computation-intensive or delay-sensitive task.

In the system, the V2V communication link, the V2I communication link, the V2V migration link and the I2I migration link are deployed at different frequencies; thus, they do not interfere with each other. There are three ways for vehicles to process their tasks: 1) process tasks locally; 2) offload tasks to idle vehicles nearby via a V2V communication link; or 3) offload tasks to the nearby RSUs for processing via a V2I communication link. Moreover, there are two ways for RSUs to migrate the computing results: 1) RSU i uses the I2I migration link to transmit the computing results to RSU q where vehicle m resides and then RSU q transmits the computing results for vehicle m through the I2V communication link; or 2) RSU i uses the I2V communication link to transmit the computing results to vehicle e in its communication range and vehicle e transmits the computing results for vehicle m through the V2V migration link.

Note that the task of vehicle m cannot be split. In the system, we use the set $\{D_{m,\text{in}}, C_{m,\text{in}}, \tau\}$ to denote the task of vehicle m. $D_{m,\text{in}}$ (in bits) is the size of the data to be processed by the task, $C_{m,\text{in}}$ (in CPU cycles) is the number of cycles required to process the task by the CPU, and τ (in s) is the maximum delay tolerance for the task. $C_{m,\text{in}} = \delta D_{m,\text{in}}$, where δ is the number of CPU cycles required for processing one bit of data. Vectors of $C_{m,\text{in}}$ and $D_{m,\text{in}}$ are denoted as $\mathbf{D_{in}} = \{D_{1,\text{in}}, D_{2,\text{in}}, \ldots, D_{M,\text{in}}\}$

and $\mathbf{C_{in}} = \{C_{1,\text{in}}, C_{2,\text{in}}, \ldots, C_{M,\text{in}}\}$. In this paper, the energy consumption of the vehicles and the delay of the system are considered. In some jobs, such as data redundancy processing, the amount of returned data is still very large, and thus cannot be ignored. Note that due to the large size of the feedback data $D_{m,\text{out}}$, the energy consumption and delay of downlink transmission must be considered. $D_{m,\text{in}} = \xi D_{m,\text{out}}$, where ξ is the reciprocal of the data processing coefficient of the RSU. For vehicle m, there are three ways to handle tasks: local processing, V2V offloading and V2I offloading. Furthermore, the system has two ways to migrate tasks, i.e., V2V migration and I2I migration. We use $\mathbf{\Psi} = \{d_m = 0, 1, 2, 3, 4\}$ to denote the offloading decision of vehicle m. Specifically, when $d_m = 0$, vehicle m processes the task locally; when $d_m = 1$, vehicle m offloads the task to an idle vehicle nearby for processing; when $d_m = 2$, vehicle m offloads the task to an RSU for processing and transfers the results directly; when $d_m = 3$, vehicle m offloads the task to an RSU for processing and uses V2V migration to transfer the computing results; and when $d_m = 4$, vehicle m offloads the task to an RSU for processing and uses I2I migration to transfer the computing results. Co-channel interference occurs when vehicles use the same subcarrier offloading tasks.

1) Local Processing: In the considered system, we assume that the vehicles are distributed according to a Poisson distribution on the highway and that the vehicles move at an average speed of $\bar{v}$. Let σ be the probability density of finding a vehicle per meter. Then, the probability of finding n vehicles in the l_{V2V}-meter lane can be expressed as

$$f(n, l_{\text{V2V}}) = \frac{(\sigma l_{\text{V2V}})^n}{n!} e^{-\sigma l_{\text{V2V}}}, \; n \geq 0. \quad (1)$$

We use L_{V2V} to express the distance between two vehicles. The probability that the distance between the vehicles is less than l_{V2V}, is written as

$$\Pr\{L_{\text{V2V}} \leq l_{\text{V2V}}\} = 1 - e^{-\sigma l_{\text{V2V}}}. \quad (2)$$

According to (2), L_{V2V} is independently identically distributed and conforms to an exponential distribution. Moreover, we can obtain the expected value $\frac{1}{\sigma}$ of the distance between the vehicles, which is treated as the average distance L_{V2V} between two vehicles, i.e., $L_{\text{V2V}} = \frac{1}{\sigma}$.

In the system, we modeled the CPU energy consumption of the vehicle as $P_{exe} = k\mu_m^3 f_{\text{ue}}^3$ as in [41], where f_{ue}, k and $\mu_m \in (0, 1]$ denote the CPU cycle frequency of vehicle m, the energy coefficient depending on the chip architecture [42] and the available rate of computational resources, respectively.

When vehicle m processes its tasks locally, the delay T_m^{local} of vehicle m is given by

$$T_m^{\text{local}} = \frac{C_{m,\text{in}}}{\mu_m f_{\text{ue}}}, \quad (3)$$

where $\mu_m \in (0, 1]$ is the available rate of computational resources. Correspondingly, the execution consumption E_m^{local} of vehicle m is expressed as

$$E_m^{\text{local}} = P_{exe} T_m^{\text{local}} = k\mu_m^2 C_{m,\text{in}} f_{\text{ue}}^2. \quad (4)$$

When we consider the weighted sum of delay and energy consumption, the local computation overhead Ω_m^{local} can be obtained as

$$\Omega_m^{\text{local}} = \alpha_m T_m^{\text{local}} + \beta_m E_m^{\text{local}}, \quad (5)$$

where α_m and β_m denote the weights of delay and energy consumption of vehicle m, respectively. $\alpha_m, \beta_m \in [0, 1], \alpha_m + \beta_m = 1$. If $\alpha_m > \beta_m$, vehicle m needs to process a delay-sensitive task; and if $\beta_m > \alpha_m$, vehicle m needs to process a computational-intensive task. When $\alpha_m = 1$ or $\beta_m = 1$, vehicle m is concerned with only the delay or energy consumption.

2) V2V Processing: V2V processing includes three main operations: task offloading, task processing and computation result returning. The vehicle offloads the task to a nearby idle vehicle; then, the idle vehicle processes the computing task and delivers the computing results to the target vehicle. If the vehicle decides to offload the task to an idle vehicle, this vehicle can offload the task via the V2V communication link. This process can lead to task-transmission delays and energy consumption. We use $R_{m,\text{tran}}^{\text{V2V}}(\boldsymbol{d})$ to denote the transmission rate of a V2V communication link. Since the same channel is used to transmit the computing result and the offloading task. It is reasonable to assume that their transmit rates are equal. Let $p_{m,\text{tran}}^{\text{V2V}}$ and $\text{b}_{m,\text{tran}}^{\text{V2V}}$ denote the transmission power and the bandwidth of the V2V communication link, respectively. According to the offloading decisions $\boldsymbol{d} = \{d_1, d_2, d_3, \ldots, d_M\}$, the transmission rate of vehicle m can be obtained. The transmission rate can be expressed as

$$R_{m,\text{tran}}^{\text{V2V}}(\boldsymbol{d}) = \text{b}_{m,\text{tran}}^{\text{V2V}} \log_2(1 + \gamma_{m,\text{tran}}^{\text{V2V}}), \quad (6)$$

where $\gamma_{m,\text{tran}}^{\text{V2V}}$ is the signal-to-interference-plus-noise ratio (SINR) of the V2V communication link. If there is no V2V communication link sharing, we have $\gamma_{m,\text{tran}}^{\text{V2V}} = p_{m,\text{tran}}^{\text{V2V}} h_{m,\text{tran}}^{\text{V2V}}/\omega$, where ω denotes the power of complex Gaussian white noise and $h_{m,\text{tran}}^{\text{V2V}}$ denotes the channel gain between adjacent vehicles that are offloading tasks through a V2V communication link. If V2V communication link sharing occurs, the SINR of the V2V communication link can be expressed as

$$\gamma_{m,\text{tran}}^{\text{V2V}} = \frac{p_{m,\text{tran}}^{\text{V2V}} h_{m,\text{tran}}^{\text{V2V}}}{\omega + \sum_{g \in \mathbf{M}, g \neq m, d_g = 1}^{M} p_{g,\text{tran}}^{\text{V2V}} h_{g,\text{tran}}^{\text{V2V}}}. \quad (7)$$

When a vehicle offloads its task to a nearby idle vehicle, we can obtain the delay $T_m^{\text{V2V}}(\boldsymbol{d})$, given by

$$T_m^{\text{V2V}}(\boldsymbol{d}) = \frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2V}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{ue}}} + \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2V}}(\boldsymbol{d})}. \quad (8)$$

Similarly, we can obtain the energy consumption $E_m^{\text{V2V}}(\boldsymbol{d})$, given by

$$E_m^{\text{V2V}}(\boldsymbol{d}) = \frac{p_{m,\text{tran}}^{\text{V2V}} D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2V}}(\boldsymbol{d})} + kC_{m,\text{in}} f_{\text{ue}}^2 + \frac{p_{m,\text{tran}}^{\text{V2V}} D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2V}}(\boldsymbol{d})}. \quad (9)$$

According to (5), we can obtain computation overhead $\Omega_m^{\text{V2V}}(\boldsymbol{d})$ of the two vehicles, given by

$$\Omega_m^{\text{V2V}}(\boldsymbol{d}) = \alpha_m T_m^{\text{V2V}}(\boldsymbol{d}) + \beta_m E_m^{\text{V2V}}(\boldsymbol{d}). \quad (10)$$

According to (10), when the number of vehicles using V2V communication links increases, the transmission rate of vehicles decrease; accordingly, the transmission time and energy consumption increase. If vehicle m needs to process a delay-sensitive task, a larger α_m can be taken. If vehicle m needs to process a computation-intensive task, a larger β_m can be taken.

3) V2I Processing: When a vehicle decides to offload its task to a nearby RSU, it can use the V2I communication link. The RSU uses its powerful computing power to process the computing task and then transmit the results back to the target vehicle. Considering that the computing result cannot be ignored, we have to calculate the resulting transmission delay and transmission energy consumption. In the system, we use $R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})$ to denote the transmission rate between vehicle m and the nearby RSU that establishes the V2I communication link. Considering the same channel is used to transmit the computing result and the offloading task, their transmit rates are equal. According to the offloading decisions $\boldsymbol{d} = (d_1, d_2, d_3, \ldots, d_M)$, the transmission rate of the V2I communication link can be obtained. The transmission rate can be expressed as

$$R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d}) = \text{b}_{m,\text{tran}}^{\text{V2I}} \log_2(1 + \gamma_{m,\text{tran}}^{\text{V2I}}), \qquad (11)$$

where $\text{b}_{m,\text{tran}}^{\text{V2I}}$ denotes the bandwidth of the V2I communication link; and $\gamma_{m,\text{tran}}^{\text{V2I}}$ is the SINR of the V2I communication link. If there is no V2I communication link sharing, $\gamma_{m,\text{tran}}^{\text{V2I}} = p_{m,\text{tran}}^{\text{V2I}} h_{m,\text{tran}}^{\text{V2I}}/\omega$, where $p_{m,\text{tran}}^{\text{V2I}}$ denotes the transmission power of the V2I communication link and $h_{m,\text{tran}}^{\text{V2I}}$ denotes the channel gain between vehicle m and the nearby RSU. If there is V2V communication link sharing, the transmission rate is affected by other vehicles that choose the same V2I communication link to offload their tasks. As a result, interference will exist during the data transmission phase. The SINR of the V2I communication link can be expressed as

$$\gamma_{m,\text{tran}}^{\text{V2I}} = \frac{p_{m,\text{tran}}^{\text{V2I}} h_{m,\text{tran}}^{\text{V2I}}}{\omega + \sum_{g\in\mathbf{M}, g\neq m, d_g=2,3,4}^{M} p_{g,\text{tran}}^{\text{V2I}} h_{g,\text{tran}}^{\text{V2I}}}. \qquad (12)$$

In the task migration model, computing tasks are all processed on the RSU, so the interference increases when $d_m = 2$, $d_m = 3$, and $d_m = 4$ are applied.

When a vehicle offloads its task to an idle RSU, we can obtain the delay $T_m^{\text{V2I}}(\boldsymbol{d})$, given by

$$T_m^{\text{V2I}}(\boldsymbol{d}) = \frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})}, \qquad (13)$$

where f_{mec} denotes the CPU cycle frequency of the RSU.

Similarly, we can obtain the energy consumption $E_m^{\text{V2I}}(\boldsymbol{d})$, given by

$$E_m^{\text{V2I}}(\boldsymbol{d}) = p_{m,\text{tran}}^{\text{V2I}} \frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + p_{m,\text{tran}}^{\text{V2I}} \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})}. \qquad (14)$$

According to (5), we can obtain computation overhead $\Omega_m^{\text{V2I}}(\boldsymbol{d})$ of vehicle m and RSU, given by

$$\Omega_m^{\text{V2I}}(\boldsymbol{d}) = \alpha_m T_m^{\text{V2I}}(\boldsymbol{d}) + \beta_m E_m^{\text{V2I}}(\boldsymbol{d}). \qquad (15)$$

4) V2V Migration: In this section, we discuss the transmission of computing results through V2V migration links. Considering some offloading tasks with a large amount of data or high time requirements, coupled with the fast speed of vehicles on the expressway, vehicles will often drive out of the RSU communication range before receiving the computing results. In this case, task migration technology is needed to transmit the computing results. The system adopts orthogonal frequency division multiple access for channel access. Thus, there is no interference between the V2V communication link and the V2V migration link. In the system, $R_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})$ denotes the migration rate between the two vehicles. According to the offloading decisions $\boldsymbol{d} = \{d_1, d_2, d_3, \ldots, d_M\}$, we can obtain the V2V migration rate. The migration rate can be expressed as

$$R_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d}) = \text{b}_{m,\text{mig}}^{\text{V2V}} \log_2(1 + \gamma_{m,\text{mig}}^{\text{V2V}}), \qquad (16)$$

where $\text{b}_{m,\text{mig}}^{\text{V2V}}$ and $\gamma_{m,\text{tran}}^{\text{V2V}}$ denote the bandwidth of the V2V migration link and the SINR of the V2V migration link, respectively. If there is no V2V migration link sharing, $\gamma_{m,\text{mig}}^{\text{V2V}} = p_{m,\text{mig}}^{\text{V2V}} h_{m,\text{mig}}^{\text{V2V}}/\omega$, where $p_{m,\text{mig}}^{\text{V2V}}$ denotes the transmission power of the V2V migration link and $h_{m,\text{mig}}^{\text{V2V}}$ denotes the channel gain between adjacent vehicles that have established the V2V migration link when transmitting computing results. If there is V2V migration link sharing, it will be interfered with by other vehicles that choose the same V2V migration link to transmit the computing results. As a result, interference will exist during the data transmission phase. The SINR of the V2V migration link can be expressed as

$$\gamma_{m,\text{mig}}^{\text{V2V}} = \frac{p_{m,\text{mig}}^{\text{V2V}} h_{m,\text{mig}}^{\text{V2V}}}{\omega + \sum_{g\in\mathbf{M}, g\neq m, d_g=3}^{M} p_{g,\text{mig}}^{\text{V2V}} h_{g,\text{mig}}^{\text{V2V}}}. \qquad (17)$$

For V2V offloading, we can obtain the delay $T_{m,\text{mig}}^{\text{V2V}}$ and the energy consumption $E_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})$, given by

$$\begin{aligned} T_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d}) &= T_m^{\text{V2I}}(\boldsymbol{d}) + \phi \frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})} \\ &= \frac{D_{m,\text{out}} + D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \phi \frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})}, \end{aligned} \qquad (18)$$

$$\begin{aligned} E_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d}) &= \frac{p_{m,\text{tran}}^{\text{V2I}} D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{p_{m,\text{tran}}^{\text{V2I}} D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} \\ &\quad + \phi \frac{p_{m,\text{mig}}^{\text{V2V}} D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})}, \end{aligned} \qquad (19)$$

where ϕ denotes the number of V2V migration vehicles that passed when the target vehicle received the computing result. Fig. 2 clearly illustrates the process of the V2V task migration. When a vehicle processes a task in V2V migration mode, the target vehicle does not receive the computing result from the RSU within the limited time. Therefore, the vehicle offloads the task to a nearby RSU through the V2I communication link. The RSU processes the task and then transmits the computing result back to a vehicle that is not the target vehicle in its coverage
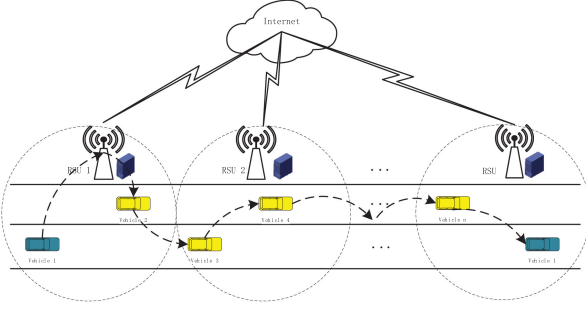
Fig. 2. The task offloading via V2V migration link.

Fig. 3. The task offloading via I2I migration link.

area. After migration, the vehicle transmits the computing result to the target vehicle.

Problem 1: When the target vehicle received the computing result, how to compute the value of ϕ that the number of V2V migrations that the target vehicle passed?

Solution 1: As shown in Fig. 2, we know that the distance of the target vehicle driven is greater than ϕ times the average distance between vehicles and less than ϕ times the average distance between vehicles plus the coverage diameter of RSU within the time when the target vehicle obtains the computing results. Thus, we can obtain

$$\phi L_{\text{V2V}} < \bar{v} T^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d}) < \phi L_{\text{V2V}} + L_{\text{I2I}},$$

$$\Rightarrow \begin{cases} \phi L_{\text{V2V}} < \bar{v}(T^{\text{V2I}}_m(\boldsymbol{d}) + \phi \frac{D_{m,\text{out}}}{R^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d})}), \\ \bar{v}(T^{\text{V2I}}_m(\boldsymbol{d}) + \phi \frac{D_{m,\text{out}}}{R^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d})}) < \phi L_{\text{V2V}} + L_{\text{I2I}}, \end{cases}$$

$$\Rightarrow \begin{cases} \phi > \dfrac{\bar{v}(\frac{D_{m,\text{in}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})}) - L_{\text{I2I}}}{L_{\text{V2V}} - \bar{v}\frac{D_{m,\text{out}}}{R^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d})}}, \\ \phi < \dfrac{\bar{v}(\frac{D_{m,\text{in}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})})}{L_{\text{V2V}} - \bar{v}\frac{D_{m,\text{out}}}{R^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d})}}, \end{cases}$$

subject to

$$L_{\text{I2I}} < \bar{v}(\frac{D_{m,\text{in}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})}),$$

$$L_{\text{V2V}} > \bar{v}\frac{D_{m,\text{out}}}{R^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d})}.$$

In the system, we express the average of $\bar{\phi}$ as

$$\bar{\phi} = \left\lfloor \frac{2\bar{v}(\frac{D_{m,\text{in}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})}) - L_{\text{I2I}}}{2(L_{\text{V2V}} - \bar{v}\frac{D_{m,\text{out}}}{R^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d})})} \right\rfloor,$$

where $\lfloor . \rfloor$ denotes rounding down.

According to (5), we can obtain computation overhead $\Omega^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d})$ of migrating computing results via the V2V migration link, as

$$\Omega^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d}) = \alpha_m T^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d}) + \beta_m E^{\text{V2V}}_{m,\text{mig}}(\boldsymbol{d}). \quad (20)$$

5) I2I Migration: In this system, we consider the communication link between adjacent RSUs. We use this link as a task migration link to transmit the computing results. In this section, we discuss the transmission of computing results via
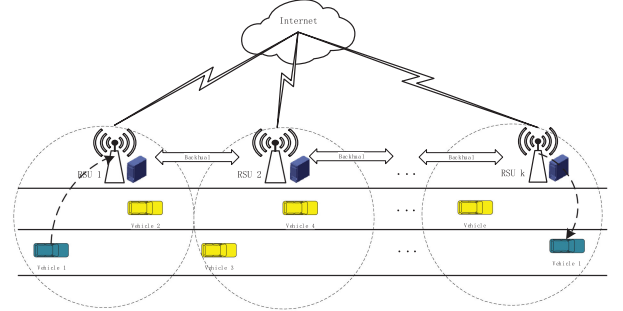
I2I migration links. Let $R^{\text{I2I}}_{m,\text{mig}}(\boldsymbol{d})$ denote the migration rate between two RSUs. According to the offloading decisions $\boldsymbol{d} = \{d_1, d_2, d_3, \ldots, d_M\}$, we can obtain the I2I migration rate. The migration rate can be expressed as

$$R^{\text{I2I}}_{m,\text{mig}}(\boldsymbol{d}) = b^{\text{I2I}}_{m,\text{mig}} \log_2(1 + \gamma^{\text{I2I}}_{m,\text{mig}}), \quad (21)$$

where $b^{\text{I2I}}_{m,\text{mig}}$ and $\gamma^{\text{I2I}}_{m,\text{tran}}$ denote the bandwidth and SINR of I2I migration link, respectively. If there is no I2I migration link sharing, $\gamma^{\text{I2I}}_{m,\text{mig}} = p^{\text{I2I}}_{m,\text{mig}} h^{\text{I2I}}_{m,\text{mig}}/\omega$, where $p^{\text{I2I}}_{m,\text{mig}}$ denotes the transmission power of the I2I migration link and $h^{\text{I2I}}_{m,\text{mig}}$ denotes the channel gain between adjacent RSUs that have established the I2I migration link when transmitting computing results. If there is I2I migration link sharing, the transmission rate will be interfered with by other vehicles that choose the same I2I migration link to transmit the computing results. As a result, interference will exist during the data transmission phase. The SINR of the I2I migration link can be expressed as

$$\gamma^{\text{I2I}}_{m,\text{mig}} = \frac{p^{\text{I2I}}_{m,\text{mig}} h^{\text{I2I}}_{m,\text{mig}}}{\omega + \sum_{g \in \mathbf{M}, g \neq m, d_g = 4}^{M} p^{\text{I2I}}_{g,\text{mig}} h^{\text{I2I}}_{g,\text{mig}}}. \quad (22)$$

For V2I offloading, we can obtain the delay $T^{\text{I2I,}}_{m,\text{mig}}$, and the energy consumption $E^{\text{I2I}}_{m,\text{mig}}(\boldsymbol{d})$, given by

$$T^{\text{I2I}}_{m,\text{mig}}(\boldsymbol{d}) = T^{\text{V2I}}_m(\boldsymbol{d}) + \varphi \frac{D_{m,\text{out}}}{R^{\text{I2I}}_{m,\text{mig}}(\boldsymbol{d})}$$

$$= \frac{D_{m,out} + D_{m,in}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \varphi \frac{D_{m,\text{out}}}{R^{\text{I2I}}_{m,\text{mig}}(\boldsymbol{d})}, \quad (23)$$

$$E^{\text{I2I}}_{m,\text{mig}}(\boldsymbol{d}) = p^{\text{V2I}}_{m,\text{tran}} \frac{D_{m,\text{in}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})} + p^{\text{V2I}}_{m,\text{tran}} \frac{D_{m,\text{out}}}{R^{\text{V2I}}_{m,\text{tran}}(\boldsymbol{d})}, \quad (24)$$

where φ denotes the number of I2I migration RSUs when the target vehicle receives the computing result. In the system, we consider only the energy consumption of vehicles: the energy consumption of transmission between RSUs is not taken into account. Fig. 3 clearly illustrates the process of I2I task migration. As in V2V task migration, when a vehicle processes a task in I2I migration mode, the vehicle offloads the task to a nearby RSU through the V2I communication link. The RSU processes the task and then transmits the computing result to an RSU that has the target vehicle in its coverage area via the I2I migration

link. After migration, the RSU transmits the computing result to the target vehicle.

Problem 2: When the target vehicle received the computing result, how to compute the value of φ that the number of I2I migrations that the target vehicle passed?

Solution 2: As shown in Fig. 3, we know that the distance of the target vehicle driven is greater than φ times the average distance between RSUs and less than φ times the average distance between RSUs plus the coverage diameter of RSU within the time when the target vehicle obtains the computing results. Therefore, we can obtain

$$\varphi L_{\text{I2I}} < \bar{v} T_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d}) < \varphi L_{\text{I2I}} + L_{\text{I2I}},$$

$$\Rightarrow \begin{cases} \varphi L_{\text{I2I}} < \bar{v}(T_m^{\text{V2I}}(\boldsymbol{d}) + \varphi \frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})}), \\ \bar{v}(T_m^{\text{V2I}}(\boldsymbol{d}) + \varphi \frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})}) < (\varphi + 1)L_{\text{I2I}}, \end{cases}$$

$$\Rightarrow \begin{cases} \varphi > \dfrac{\bar{v}(\frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})}) - L_{\text{I2I}}}{L_{\text{I2I}} - \bar{v}\frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})}}, \\ \varphi < \dfrac{\bar{v}(\frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})})}{L_{\text{I2I}} - \bar{v}\frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})}}, \end{cases}$$

subject to

$$L_{\text{I2I}} < \bar{v}(\frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})}),$$

$$L_{\text{I2I}} > \bar{v}\frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})}.$$

In the system, we express the average of $\bar{\varphi}$ as

$$\bar{\varphi} = \left\lfloor \frac{2\bar{v}(\frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})}) - L_{\text{I2I}}}{2(L_{\text{I2I}} - \bar{v}\frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})})} \right\rfloor,$$

where $\lfloor . \rfloor$ denotes rounding down.

According to (5), we can obtain computation overhead $\Omega_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})$ of migrating computing results via the I2I migration link as

$$\Omega_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d}) = \alpha_m T_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d}) + \beta_m E_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d}). \qquad (25)$$

B. Problem Formulation

In this section, according to the task offloading decisions $\boldsymbol{d}_{-m} \triangleq (d_1, \ldots, d_{m-1}, d_{m+1}, \ldots, d_M)$ of all other vehicles except vehicle m, we analyze the impact of the offloading decision of vehicle m on the overall computation overhead.

1) Minimize the Computation Overhead of the System: In this section, we discuss how to minimize the computation overhead of the whole network. According to the discussion in Section III, vehicle m has three alternative ways to process its task and two alternative ways to migrate the computing result. The choice of vehicle m directly affects the computation overhead of the whole network. As the number of tasks in the V2V communication link, V2I communication link, V2V migration link and I2I migration link increases, their transmission rates decrease, thereby increasing the transmission delay and energy consumption. As a result, the computation overhead of

the whole network increases, which is contrary to our purpose. In this paper, we design an offloading scheme under various constraints to minimize the computation overhead $\Upsilon(\boldsymbol{d})$ of the whole system. Under the given tasks offloading decisions $\boldsymbol{d} = \{d_1, d_2, d_3, \ldots, d_M\}$ of all other vehicles, we can obtain the computation overhead $\Upsilon(\boldsymbol{d})$ of the network which is described as a mathematical problem, as given by

$$\mathbf{P1}: \min_{\boldsymbol{d}, \tau, \alpha_m, \beta_m, \mu_m} \Upsilon(\boldsymbol{d}) = \sum_{m=1}^{M} (\Gamma_{\{d_m=0\}} \Omega_m^{\text{local}}$$

$$+ \Gamma_{\{d_m=1\}} \Omega_m^{\text{V2V}} + \Gamma_{\{d_m=2\}} \Omega_m^{\text{V2I}}$$

$$+ \Gamma_{\{d_m=3\}} \Omega_{m,\text{mig}}^{\text{V2V}} + \Gamma_{\{d_m=4\}} \Omega_{m,\text{mig}}^{\text{I2I}}), \qquad (26)$$

$$\text{s.t.} \quad C1: \quad 0 \le p_{m,\text{tran}}^{\text{V2V}} \le p_{m,\text{tran}}^{\text{V2V,max}}, \forall m \in \mathbf{M},$$

$$C2: \quad 0 \le p_{m,\text{tran}}^{\text{V2I}} \le p_{m,\text{tran}}^{\text{V2I,max}}, \forall m \in \mathbf{M},$$

$$C3: \quad 0 \le p_{m,\text{mig}}^{\text{V2V}} \le p_{m,\text{mig}}^{\text{V2V,max}}, \forall m \in \mathbf{M},$$

$$C4: \quad 0 \le p_{m,\text{mig}}^{\text{I2I}} \le p_{m,\text{mig}}^{\text{V2V,max}}, \forall m \in \mathbf{M},$$

$$C5: \quad \sum_{m=1}^{M} \Gamma_{\{d_m=g\}} \le M, \forall m \in \mathbf{M}, g \in \{0, 1, 2, 3, 4\},$$

$$C6: \quad d_m \in \Psi, \forall m \in \mathbf{M},$$

$$C7: \quad T_m^{\text{local}} \le \tau, \forall m \in \mathbf{M},$$

$$C8: \quad T_m^{\text{V2V}}(\boldsymbol{d}) \le \tau, \forall m \in \mathbf{M},$$

$$C9: \quad T_m^{\text{V2I}}(\boldsymbol{d}) \le \tau, \forall m \in \mathbf{M},$$

$$C10: \quad T_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d}) \le \tau, \forall m \in \mathbf{M},$$

$$C11: \quad T_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d}) \le \tau, \forall m \in \mathbf{M},$$

$$C12: \quad \alpha_m + \beta_m = 1, \forall m \in \mathbf{M},$$

$$C13: \quad \mu_m \in (0, 1],$$

where $\Gamma_{\{d_m=g\}}$ is an indicator function. If $d_m = g$ is true, $\Gamma_{\{d_m=g\}} = 1$; otherwise, $\Gamma_{\{d_m=g\}} = 0$.

In P1, C1, C2, C3 and C4 are the maximum transmission power constraints of the V2V communication link, V2I communication link, V2V migration link and I2I migration link, respectively. C5 denotes the constraint of the number of vehicles that use five modes to process the computing task. C6 denotes the offloading decision choice of vehicle m. C7, C8, C9, C10 and C11 are the maximum transmission delay constraints of local processing, V2V processing, V2I processing, V2V migrating and I2I migrating, respectively. C12 indicates the relationship between the weight of delay and the weight of energy consumption. C13 denotes the constraint of the available rate of computational resources.

When the target vehicle uses task migration, the following constraint settings must be met, as given by

$$\text{s.t.} \quad S1: \quad L_{\text{I2I}} < \bar{v}(\frac{D_{m,\text{in}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})} + \frac{C_{m,\text{in}}}{f_{\text{mec}}} + \frac{D_{m,\text{out}}}{R_{m,\text{tran}}^{\text{V2I}}(\boldsymbol{d})}),$$

$$\forall m \in \mathbf{M},$$

$$S2: \quad L_{V2V} > \bar{v}\frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{V2V}(\boldsymbol{d})}, \forall m \in \mathbf{M},$$

$$S3: \quad L_{I2I} > \bar{v}\frac{D_{m,\text{out}}}{R_{m,\text{mig}}^{I2I}(\boldsymbol{d})}, \forall m \in \mathbf{M},$$

where S1 indicates that the vehicle must be driven out of RSU communication range within a limited time to use the migration. S2 indicates that when a vehicle adopts V2V migration to transmit the computing result, the distance that the vehicle drives in a single V2V migration time should be less than the average distance between two vehicles; otherwise, the vehicle will never catch up with the computing result. Similarly, S3 indicates that when a vehicle adopts I2I migration to transmit the computing result, the distance that the vehicle drives in a single I2I migration time should be less than the average distance between two RSUs; otherwise, the vehicle will never catch up with the computing result.

In the following sections, we provide a game theory-based strategy to solve the problem of offloading decisions among vehicles. This strategy can effectively reduce the weighted sum of the delay and energy consumption of the system.

2) Minimize the Computation Overhead of the Vehicle: In this section, we discuss how to minimize the computation overhead of the vehicle. The computation offloading decisions $\boldsymbol{d}$ of the vehicles are coupled in the system. When a large number of vehicles choose the same link to offload tasks or to migrate the computing results, they can cause interference between tasks and slow the transmission and migration rates. Even very low transmission rates and migration rates can increase transmission delays, transmission energy consumption, migration delays and migration energy consumption. In this case, it is practical and efficient for the vehicles to choose local processing of computation tasks with task migration to avoid the long processing delay and high energy consumption. V2V migration and I2I migration can be reasonably used for tasks requiring computing migration.

Under the given task offloading decisions $\boldsymbol{d}_{-m} \triangleq (d_1, \ldots, d_{m-1}, d_{m+1}, \ldots, d_M)$ of all other vehicles except vehicle m, we obtain the computation overhead function of vehicle m as

$$\Omega_m(d_m, \boldsymbol{d}_{-m}) = \begin{cases} \Omega_m^{\text{local}}, & if\ d_m = 0, \\ \Omega_m^{V2V}, & if\ d_m = 1, \\ \Omega_m^{V2I}, & if\ d_m = 2, \\ \Omega_{m,\text{mig}}^{V2V}, & if\ d_m = 3, \\ \Omega_{m,\text{mig}}^{I2I}, & if\ d_m = 4. \end{cases} \quad (27)$$

In the following sections, we provide a game theory-based strategy to solve the problem of offloading decisions among vehicles to effectively reduce the weighted sum of the delay and energy consumption of the vehicle.

IV. COMPUTATION OFFLOADING AND TASK MIGRATION

In this section, we provide a game approach for the considered computation offloading system, and then formulate the offloading decision-making problem as a game.

Vehicles belong to rational (or selfish) people whose goal is to prevent damage to their own interests, and game theory considers how selfish players make decisions. Thus, we choose game theory as our solution. In game theory, because people are rational (or selfish), each person chooses the decision that minimizes his or her own computation overhead, regardless of the positive or negative impact of his or her decision on others. Game theory is an effective method for vehicles to analyze their interests and obtain the minimum computation overhead through offloading decisions.

A. Game Formulation

Different vehicles have different interests and capabilities, and run different applications. Analyzing the interactions among multiple vehicles by means of game theory is convenient. Specifically, we consider that vehicles choose their offloading decisions sequentially. We formulate the offloading decision-making problem as a game. Game theory has the advantage of not only helping to develop low-complexity algorithms, but also reducing the amount of computation required by algorithms.

By changing the offloading decision of vehicle m, we can obtain the minimized weighted sum of delay and energy consumption of all vehicles within a finite number of iterations. The weighted sum of delay and energy consumption is related not only to the offloading strategy of vehicle m but also to the offloading decisions of other vehicles. This is because when the offloading decision of vehicle m changes, the transmission rate or migration rate of other vehicles may change, thus affecting the computation overhead of other vehicles. Given offloading decisions $\boldsymbol{d}_{-m} \triangleq (d_1, \ldots, d_{m-1}, d_{m+1}, \ldots, d_M)$ of other vehicles, vehicle m can choose the best offloading decision d_m to minimize its own computation overhead, i.e.,

$$\min_{d_m \in \{0,1,2,3,4\}} \Omega_m(d_m, \boldsymbol{d}_{-m}), \ \forall m \in \mathbf{M}.$$

When the offloading decisions of other vehicles are known, vehicle m can select the offloading decision with the lowest computation overhead by comparing the computation overhead under the five offloading decisions. Then, we describe the above problem as the offloading decision game $\Delta = (\mathbf{M}, \boldsymbol{\Psi}, \{\Omega_m\}_{m \in \mathbf{M}})$, where $\mathbf{M}$ is the finite set of the player, $\boldsymbol{\Psi}$ is the set of strategies for vehicle m, and $\{\Omega_m\}_{m \in \mathbf{M}}$ is the set of computation overhead functions of vehicle m. In game theory, the NE is an important point. Next, we will introduce the NE.

Theorem 1: If only one player that has finite action sets makes a decision at a time and knows every action of other players, we call the game Δ a perfect information sequential offloading game. A finite sequential game with perfect information exists an NE, and it can be converged within a finite number of decision slots, which is called the FIP.

Proof: The proof is given in [43]. ∎

Definition 1: For the sequential offloading game Δ, if no player can change his/her offloading decision to reduce the overall computation overhead under given offloading decision $\boldsymbol{d}_{-m} \triangleq (d_1, \ldots, d_{m-1}, d_{m+1}, \ldots, d_M)$, the offloading decision

$\boldsymbol{d}^*$ is the Nash equilibrium of the game if and only if the following conditions are satisfied:

$$\Omega_m(d_m^*, \boldsymbol{d}_{-m}^*) \leq \Omega_m(d_m, \boldsymbol{d}_{-m}^*), \ \forall m \in \mathbf{M}, \ \forall d_m \in \boldsymbol{\Psi}.$$

According to the FIP, after a certain number of iterations, the game theory will move towards the NE which is stable. It is means that, under the given set of offloading decisions $\boldsymbol{d}_{-m} \triangleq (d_1, \ldots, d_{m-1}, d_{m+1}, \ldots, d_M)$ of other vehicles, vehicle m cannot reduce the total computation overhead of all vehicles by changing its own offloading decision. Therefore, based on the Nash equilibrium principle of game theory, each vehicle can obtain its own satisfactory offloading decision.

B. Convergence

In this section, we will discuss the convergence of game theory. When a finite number of players participate in a game, and each player has a finite number of decisions to choose, the game can reach NE in a finite number of steps [25], [44], according to the existence theorem of NE.

Theorem 2: Starting at any offloading decision, the offloading decision-making game of Algorithm 2 is guaranteed to eventually converge.

Proof: See Appendix A. ∎

C. Complexity

Each cycle performs M iterations to calculate the minimum computation overhead per vehicle. When Algorithm 2 performs Q cycles to reach NE, the complexity of the algorithm is $O(MQ)$, which is much lower than the complexity of the exhaustive array of $O(5^M)$.

D. Task Migration

In this section, we consider the case of using task migration when the amount of computations is large and the speed is fast.

In the MEC-assisted vehicular network, the RSU has two links to transmit the computing result, i.e., the V2V migration link and the I2I migration link. According to the discussion in Section III, considering the co-channel interference, when too many tasks use the same migration link to transmit the computing results, the migration rate will be slow, and the migration delay and migration energy consumption will increase. Therefore, in this subsection, we make reasonable use of migration links and transmission links to minimize the vehicle computation overhead. Next, we introduce the migration selection algorithm and the specific steps of the migration selection algorithm, as shown in Algorithm 1.

E. Computation Overhead Minimization Offloading Algorithm

Based on the above study, we propose a task migration (TM) algorithm based on game theory and Algorithm 1. In the TM algorithm, the offloading decision with the lowest computation overhead is determined by changing the offloading decision of the vehicle. Using the above approach, after a period of iteration, the game will reach NE, and each vehicle makes the offloading

---

Algorithm 1: Migration Selection (MS) Algorithm.

1) **Input**: $f_{\text{ue}}, \mu_m, k, f_{\text{mec}}$. Task $(D_{m,\text{in}}, C_{m,\text{in}}, \tau), \alpha_m,$ β_m, vehicle $m\{m = 1, 2, 3 \ldots M\}$.
2) **Output**: offloading decisions $\boldsymbol{d} = (d_1, d_2, d_3, \ldots, d_M)$ and the computation overhead $\Upsilon(\boldsymbol{d})$.
3) **initialization**: offloading decisions of all vehicles are set as 0, i.e., $d_m = 0, \forall m \in \mathbf{M}$, so the set of offloading decisions is $\boldsymbol{d} = (0_1, 0_2, 0_3, \ldots, 0_M)$ in the beginning.
4) **begin**:
5)   vehicle m generates computing task $(D_{m,\text{in}}, C_{m,\text{in}}, \tau)$;
6)   vehicle m offloads task to the nearby RSU s;
7)   the RSU s processes the task;
8)   **if** vehicle m is still in the communication range of RSU s when RSU s has processed the task **then**
9)   let vehicle m receive the computing results directly from RSU s via a V2I communication link;
10)   calculate the computation overhead $\Omega_m^{\text{V2I}}(\boldsymbol{d})$ of vehicle m;
11)   **else**
12)   let the computing results be transmitted via a V2V migration link or an I2I migration link;
13)   after migrating, let vehicle m receive the computing result eventually;
14)   calculate the computation overhead $\Omega_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})/\Omega_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})$ of vehicle m;
15)   **end if**
16) **end**

---

decision that minimizes its own computation overhead. Considering the task offload situation, for each vehicle, the TM algorithm achieves the minimum value of its own computation overhead on the condition that the offloading decisions of other vehicles are known. The detailed procedure of the proposed algorithm is shown in Algorithm 2.

In Algorithm 2, according to game theory, each vehicle chooses the offloading decision that minimizes its own computation overhead. After a finite number of iterations, the game reaches NE, and the vehicle reaches the offloading decision that minimizes its own computation overhead under the condition that the offloading decisions of other vehicles are known. In this equilibrium state, the computation overhead for each vehicle is minimal, but the computation overhead for the whole system may not reach the minimum value. Therefore, the offloading decisions obtained by game theory may not be optimal for the computation overhead of the system. To minimize the computation overhead of the whole system, we propose a computation overhead minimization offloading (COMO) algorithm, whose detailed steps are described in Algorithm 3.

In Algorithm 3, the best offloading direction is similar to the steepest descent direction in reducing the computation overhead of the whole system. If $\Upsilon(\boldsymbol{d}_t) \not< \Upsilon(\boldsymbol{d}_{t+1})$, the offloading decision of the vehicle will be updated in the next time slot; otherwise, the vehicle will keep the offloading decision unchanged in the next time slot, i.e., $d_{m,t+1} = d_{m,t}$. For the offloading decision updating problem, we set the number of vehicles as the number

---

Algorithm 2: Task Migration (TM) Algorithm.

1) **Input**: f_{ue}, μ_m, k, f_{mec}. Task $(D_{m,\text{in}}, C_{m,\text{in}}, \tau)$, α_m, β_m, vehicle $m\{m = 1, 2, 3 \ldots M\}$.
2) **Output**: offloading decisions $\boldsymbol{d} = (d_1, d_2, d_3, \ldots, d_M)$ and the computation overhead $\Upsilon(\boldsymbol{d})$.
3) **initialization**: offloading decisions of all vehicles are set as 0, i.e., $d_m = 0$, $\forall m \in \mathbf{M}$, so the set of offloading decisions is $\boldsymbol{d} = (0_1, 0_2, 0_3, \ldots, 0_M)$ in the beginning.
4) **begin** :
5) let vehicle m generate a computing task;
6) **repeat**
7) **for** $m \in \mathbf{M}$ **do**
8)   calculate the rates based on known offloading decision $\boldsymbol{d}$;
9)   calculate T_m^{local}, $T_m^{\text{V2V}}(\boldsymbol{d})$, $T_m^{\text{V2I}}(\boldsymbol{d})$, $T_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})$ and $T_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})$ according to (3), (8), (13), (18) and (23);
10)   **if** all the delays are greater than τ **then**
11)   the task cannot be completed on time;
12)   **else**
13)   calculate Ω_m^{local}, $\Omega_m^{\text{V2V}}(\boldsymbol{d})$, $\Omega_m^{\text{V2I}}(\boldsymbol{d})$, $\Omega_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})$ and $\Omega_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})$ according to (5), (10), (15), (20) and (25);
14)   take the minimum of Ω_m^{local}, $\Omega_m^{\text{V2V}}(\boldsymbol{d})$, $\Omega_m^{\text{V2I}}(\boldsymbol{d})$, $\Omega_{m,\text{mig}}^{\text{V2V}}(\boldsymbol{d})$ and $\Omega_{m,\text{mig}}^{\text{I2I}}(\boldsymbol{d})$ whose delay is less than τ;
15)   calculate d_m, which is the minimum value;
16)   **end if**
17) **end for**
18) obtain the offloading decision $\boldsymbol{d}^*$;
19) **if** $\boldsymbol{d} \neq \boldsymbol{d}^*$ **then**
20)   send update-request (UR) to the server;
21)   **if** receive the permission of UR **then**
22)   obtain the offloading decision $\boldsymbol{d}^*$;
23)   update the offloading decision $\boldsymbol{d} = \boldsymbol{d}^*$;
24)   **end if**
25) **end if**
26) **until** no UR for the offloading decision.
27) calculate the computation overhead $\Upsilon(\boldsymbol{d})$;
28) **return** offloading decision $\boldsymbol{d}$ and computation overhead $\Upsilon(\boldsymbol{d})$.
29) **end**

Algorithm 3: Computation Overhead Minimization Offloading (COMO) Algorithm.

1) **Input**: f_{ue}, k, μ_m, f_{mec}, $(D_{m,\text{in}}, C_{m,\text{in}}, \tau)$, α_m, β_m, vehicle $m\{m = 1, 2, 3 \ldots M\}$.
2) **Output**: offloading decision $\boldsymbol{d} = (d_1, d_2, d_3, \ldots, d_M)$, computation overhead $\Upsilon(\boldsymbol{d})$.
3) **initialization**: each vehicle m chooses the offloading decision $d_m = 0$, $\forall m \in \mathbf{M}$.
4) **begin**
5) let vehicle m generate a computing task;
6) **repeat**
7) **for** $m \in \mathbf{M}$ **do**
8)   obtain offloading decision $\boldsymbol{d}$;
9)   **for** each time slot t;
10)   offloading decision $d_{m,t} \to d_{m,t+1}$;
11)   computation overhead $\Upsilon(\boldsymbol{d}_t) \to \Upsilon(\boldsymbol{d}_{t+1})$;
12)   obtain $\boldsymbol{d}_{t+1}^*$;
13)   **if** $\Upsilon(\boldsymbol{d}_t) < \Upsilon(\boldsymbol{d}_{t+1})$**then**
14) $\boldsymbol{d}_{t+1}^* = \boldsymbol{d}_{t+1}^*$;
15)   **else**
16) $\boldsymbol{d}_{t+1}^* \neq \boldsymbol{d}_t^*$;
17)   **end if**
18)   **end for**
19)   obtain the offloading decision $\boldsymbol{d}^*$;
20)   **if** $\boldsymbol{d} \neq \boldsymbol{d}^*$ **then**
21)   the server asks the vehicles to update their offloading decisions;
22)   update the offloading decision $\boldsymbol{d} = \boldsymbol{d}^*$;
23)   **end if**
24) **end for**
25) **until** the server does not send an update request.
26) **return** offloading decision $\boldsymbol{d}$ and computation overhead $\Upsilon(\boldsymbol{d})$.

---

of iterations of each cycle. When the offloading decision of each vehicle does not change at the end of each cycle iteration (i.e., the computation overhead of the whole system does not change) or the server does not send out the offloading decision update request, the offloading decision is optimal.

V. SIMULATION RESULTS

In this section, we provide simulation results to demonstrate the performance of the proposed algorithms in the MEC-assisted vehicular network system. RSUs are located on one-way roads. We use urban mobility simulations to simulate road traffic. The speed of vehicles is $108\,\text{km/h}$, and the maximum delay

tolerance is τ for the task. The simulation parameters are shown in Table I. Based on the most basic no-migration task offloading scheme MEC to evaluate the efficiency of the TM algorithm and the COMO algorithm, we also compare the proposed algorithm with the VCMO algorithm [46].

Fig. 4 shows the computation overhead of the whole system and each vehicle under different numbers of vehicles. In this simulation, we do not consider the maximum delay tolerance τ. Fig. 4 shows that the computation overhead of the whole system and that of each vehicle increases as the number of vehicles increases. For the TM algorithm and the COMO algorithm, the computation overhead of the whole system and that of each vehicle are significantly lower than those for the MEC and VCMO algorithms. In Fig. 4(a), initially, the computation overhead of the VCMO algorithm, the TM algorithm and the COMO algorithm are basically the same. As the number of vehicles increases, the computation overhead gap between the COMO algorithm and the VCMO algorithm becomes large. In Fig. 4(b), when the number of vehicles is small, the vehicle can offload the task to a nearby RSU, and task migration is adopted for the large amount of data; thus, the computation overhead

TABLE I
SIMULATION PARAMETERS

| Parameter	Value	Parameter	Value
Distance between adjacent vehicles L_{V2V}	$100\,m$	The available rate of computational resources μ	$0.5 \sim 1$
Distance between adjacent RSUs L_{I2I}	$300\,m$	The power of complex Gaussian white noise ω	10^{-10}
Distance between RSU and vehicle L_{V2I}	$100\,m$	The transmission power of the V2I communication link $p_{m,\mathrm{tran}}^{\mathrm{V2I}}$	$400\,\mathrm{mW}$
The size of data to be processed $D_{m,\mathrm{in}}$	$5\,\mathrm{Mbit} \sim 50\,\mathrm{Mbit}$	The bandwidth of the V2I communication link $b_{m,\mathrm{tran}}^{\mathrm{V2I}}$	$10\,\mathrm{MHz}$
The number of cycles by the CPU $C_{n,\mathrm{in}}$	$\delta D_n\ cycles$	The channel gain of the V2I communication link $h_{m,\mathrm{tran}}^{\mathrm{V2I}}$	$L_{\mathrm{V2I}}^{-\theta}$
δ	$3.055 * 10^3$	The bandwidth of the V2V migration link $b_{m,\mathrm{mig}}^{\mathrm{V2V}}$	$4\,\mathrm{MHz}$
The size of the feedback data $D_{m,\mathrm{out}}$	$D_{m,\mathrm{in}} = \xi D_{m,\mathrm{out}}$	The transmission power of the V2V migration link $p_{m,\mathrm{mig}}^{\mathrm{V2V}}$	$100\,\mathrm{mW}$
ξ	100	The channel gain of the V2V migration link $h_{m,\mathrm{mig}}^{\mathrm{V2V}}$	$L_{\mathrm{V2V}}^{-\theta}$
The CPU cycle frequency of vehicle f_{ue}	$1\,\mathrm{GHz}$	The bandwidth of the I2I migration link $b_{m,\mathrm{mig}}^{\mathrm{I2I}}$	$4\,\mathrm{MHz}$
The CPU cycle frequency of RSU f_{mec}	$5\,\mathrm{GHz}$	The transmission power of the I2I migration link $p_{m,\mathrm{mig}}^{\mathrm{I2I}}$	$400\,\mathrm{mW}$
The energy coefficient k	10^{-28} [45]	The channel gain of the I2I migration link $h_{m,\mathrm{mig}}^{\mathrm{I2I}}$	$L_{\mathrm{I2I}}^{-\theta}$
The weights of delay α_m	$0 \sim 1$	The transmission power of the V2V communication link $p_{m,\mathrm{tran}}^{\mathrm{V2V}}$	$100\,\mathrm{mW}$
The weights of energy consumption β_m	$0 \sim 1$	The bandwidth of the V2V communication link $b_{m,\mathrm{tran}}^{\mathrm{V2V}}$	$10\,\mathrm{MHz}$
The path loss exponent θ	4	The channel gain of the V2V communication link $h_{m,\mathrm{tran}}^{\mathrm{V2V}}$	$L_{\mathrm{V2V}}^{-\theta}$
The average speed $\bar{v}$	$108\,\mathrm{km/h}$	The average number of vehicles that pass by V2V migration ϕ	
The maximum delay tolerance for the task τ		The average number of RSUs that pass by I2I migration $\bar{\varphi}$	
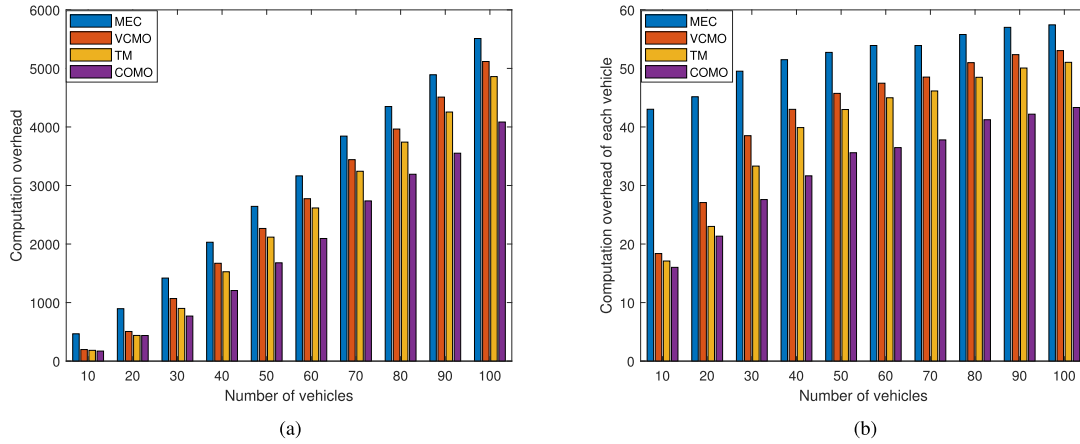
Fig. 4. Computation overhead under different numbers of vehicles. (a) Computation overhead of whole system. (b) Computation overhead of each vehicle.

TABLE II
REDUCTION RATE

	Average computation overhead reduction rate			Average delay reduction rate			Average energy consumption reduction rate				
	5Mbits	15Mbits	50Mbits	5Mbits	15Mbits	50Mbits	5Mbits	10Mbits	15Mbits	20Mbits	50Mbits
VCMO	2.3%	13.0%	7.7%	2.2%	9.4%	5.7%	0.0%	−1.3%	0.3%	0.8%	0.8%
TM	3.3%	15.5%	12.9%	2.8%	12.1%	10.0%	0.0%	−1.3%	0.7%	1.7%	0.1%
COMO	3.6%	24.6%	22.6%	6.7%	18.7%	17.1%	4.3%	2.8%	2.4%	5.0%	5.0%

of each vehicle is low. As the number of vehicles increases, the co-channel interference on offloading and migration links improves the overall computation overhead, resulting in an increase in the computation overhead of each vehicle.

Fig. 5(a), (b) and (c) show the average computation overhead, average delay and average energy consumption versus the data size. We do not consider the maximum delay tolerance τ. From Fig. 5(a), (b) and (c), we can clearly see that as the data size increases, the average computation overhead, the average delay and the average energy consumption increases accordingly. Moreover, the COMO algorithm, TM algorithm and VCMO algorithm, which take into account task migration, perform better than MEC without considering task migration. When the data size is large, the COMO algorithm performs

best. Fig. 5(d), (e) and (f) show the average computation overhead reduction rate, the average delay reduction rate and the average energy consumption reduction rate versus the data size. In the system, we define the reduction rate as $(\Upsilon_{\mathrm{mec}}(\boldsymbol{d}) - \Upsilon_{\mathrm{VCMO/TM/COMO}}(\boldsymbol{d}))/\Upsilon_{\mathrm{mec}}(\boldsymbol{d})$. Some reduction rate data are shown in Table II. Initially, the size of the vehicle computing task data is small, the number of vehicles applying task migration is small, and the reduction rate of the average computation overhead is low. As the vehicle computing task data size increases, more vehicles need to use task migration, and the reduction rate of the average computation overhead increases rapidly from 3.6% to 24.6%. When the vehicle computing task data size is larger than 15 Mbits, the reduction rate of the average computation overhead slowly decreases. Accordingly, the average
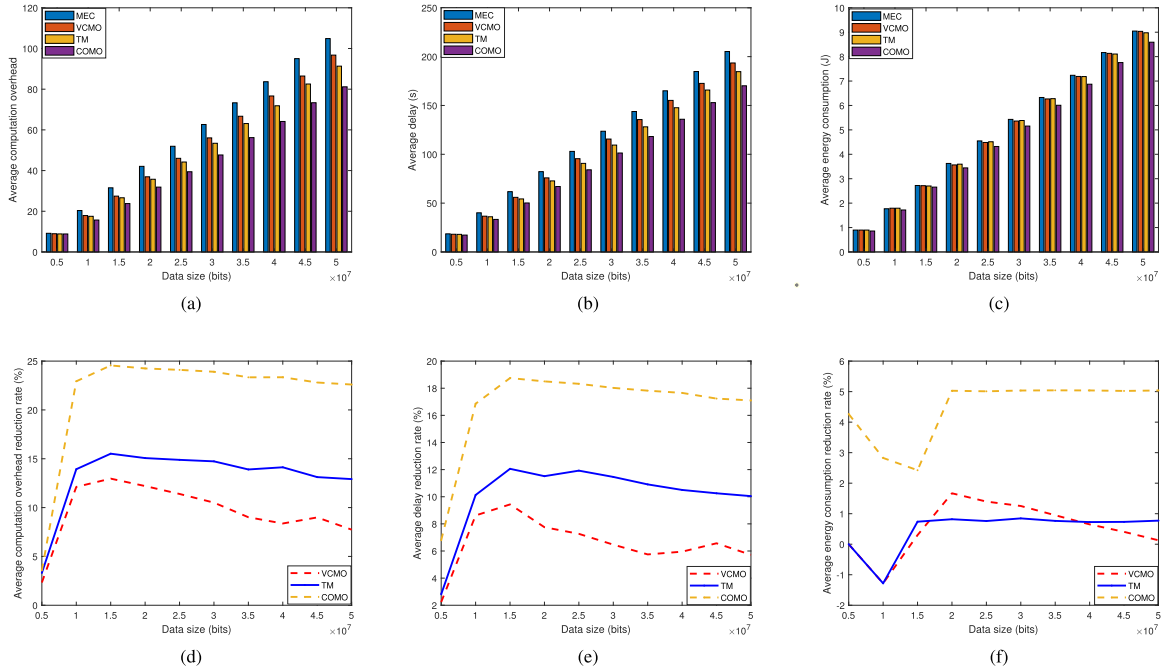
Fig. 5. Performance of different algorithms with different data sizes with $M = 100$. (a) Average computation overhead. (b) Average delay. (c) Average energy consumption. (d) Average computation overhead reduction rate. (e) Average delay reduction rate. (f) Average energy consumption reduction rate.
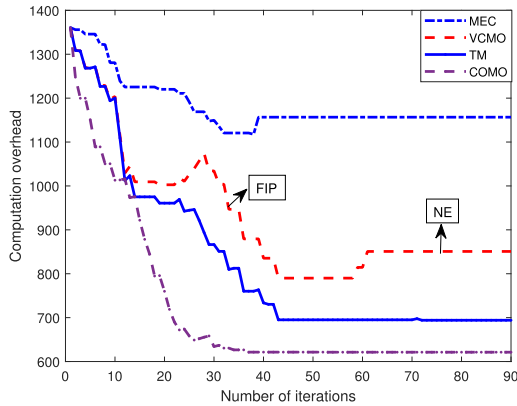

Fig. 6. Convergence behavior of different algorithms in terms of the system overhead with $M = 30$.

delay reduction rate increases from 6.7% to 18.7%, and the average energy computation reduction rate increases from 4.3% to 5.0%. In Fig. 5(f), the curve of the average energy consumption reduction rate shows an initial decline in the rising trend. This is because if we consider only the energy consumption of vehicles, when the data size is in a certain range, the local energy consumption is slightly lower than other forms of energy consumption, but migration tasks cause the energy consumption of TM and VCMO to be higher than that of MEC. As the data size increases, local energy consumption exceeds other forms of energy consumption. Then, the energy consumption of TM and VCMO is lower than that of MEC.

Fig. 6 shows the convergence behavior of different algorithms when the system has 30 vehicles. In this simulation, we do not

consider the maximum delay tolerance τ for the task. Based on the convergence and NE of the game, and according to the FIP property, the game can go through a finite number of steps to reach the NE. Clearly, all vehicles choose to process tasks locally initially, so this is the point where the computation overhead is highest. As the game proceeds, more tasks are offloaded to nearby vehicles and RSUs, and the computing results are migrated via V2V or I2I migration links. Some bulges are observed on the VCMO and TM curves because in the system, the vehicle is seen as the player and every vehicle is aiming to minimize its own computation overhead. Even if the vehicle minimizes its own computation overhead, its offloading decision may result in an increase in the transfer rate or migration rate, which in turn increases the overall computation overhead, resulting in a bump in the curve. After a finite number of iterations, the game reaches NE. Furthermore, as the game reaches NE, MEC has the highest computation overhead, followed by VCMO, TM and COMO.

Fig. 7 shows the success rate of different algorithms with a time limit. The success rate refers to the percentage of tasks completed within the time limit. We set the variable μ_m randomly as $[0.5, 1]$ and α_m is randomly assigned from $\{0, 0.5, 1\}$ for vehicle m. The number of vehicles is set to 100, that is, $M = 100$. The success rate of different algorithms increases almost linearly with an increase in the time limit. When the time limit is short, the success rates of the VCMO algorithm, TM algorithm and COMO algorithm are very low and tend to be the same due to the large size of the computing task. Compared with the VCMO algorithm, within a certain range of time limits, the TM algorithm and COMO algorithm have high success rates. As the time limit increases, the success rates of different algorithms tend to be the same. When the success rate is 100%, the value
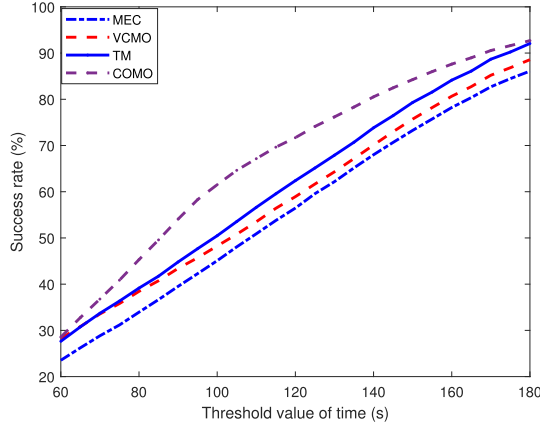
Fig. 7. Success rate of different algorithms with limited time threshold.

of the maximum delay tolerance and the threshold value of time in Fig. 7 are the same.

VI. CONCLUSION

In this paper, we have studied an MEC-assisted vehicular network that considers task migration. In the system, we proposed a TM algorithm to minimize vehicle computation overhead and a COMO algorithm to minimize system computation overhead. We have also considered the large amount of computing task data and the vehicle driving out of the RSU communication range at fast speed. By means of a joint task migration and task offloading model, we have modeled the offload decision problem as a game and have demonstrated that the game always has an NE. Moreover, both algorithms can reach the NE in a finite number of steps. Numerical simulations have shown that the proposed TM and COMO algorithms can reduce the computation overhead and improve the success rate.

APPENDIX A
PROOF OF THEOREM 1

Proof: To reduce the computation overhead of the vehicle itself, the vehicle selects the offloading scheme with the minimum computation overhead from five offloading schemes. Consider two consecutive iteration states i and $i + 1$, and assume that the offloading decision $d_{m,i+1}$ was formed from $d_{m,i}$ after an iteration was performed. Both operations occur if and only if the vehicle computation overhead is decreased. It can be expressed as

$$d_{m,i} \rightarrow d_{m,i+1} \Leftrightarrow \Omega_m(d_{m,i}, \boldsymbol{d}_{-m}) > \Omega_m(d_{m,i+1}, \boldsymbol{d}_{-m}).$$

Therefore, the computation overhead of vehicle m is always decreasing, as shown by

$$d_{m,initial} \rightarrow d_{m,1} \rightarrow d_{m,2} \rightarrow \cdots \rightarrow d_{m,final},$$

where $d_{m,initial}$ and $d_{m,final}$ are the initial and final decisions of vehicle m, respectively. Therefore, the computation overhead per vehicle is minimized. Since the number of vehicles is finite and the offloading decisions of vehicles are also finite, the

Nash equilibrium can be guaranteed under a finite number of iterations. ∎

REFERENCES

[1] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Veh. Technol. M*, vol. 12, no. 2, pp. 36–44, Jun. 2017.

[2] Q. Cui *et al.*, "Big Data analytics and network calculus enabling intelligent management of autonomous vehicles in a smart city," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2021–2034, Apr. 2019.

[3] J. Wang *et al.*, "Vehicular sensing networks in a smart city: Principles, technologies and applications," *IEEE Trans. Veh. Technol.*, vol. 25, no. 1, pp. 122–132, Feb. 2018.

[4] H. Wang, X. Li, H. Ji, and H. Zhang, "Federated offloading scheme to minimize latency in MEC-enabled vehicular networks," in *Proc. IEEE Globecom Workshops*, Abu Dhabi, United Arab Emirates, 2018, pp. 1–6.

[5] Q. Li, Y. Andreopoulos, and M. van der Schaar, "Streaming-viability analysis and packet scheduling for video over in-vehicle wireless networks," *IEEE Trans. Veh. Technol.*, vol. 56, no. 6, pp. 3533–3549, Nov. 2007.

[6] S. Zhang, Y. Hou, X. Xu, and X. Tao, "Resource allocation in D2D-based V2V communication for maximizing the number of concurrent transmissions," in *Proc. IEEE Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, Valencia, Spain, 2016, pp. 1–6.

[7] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011.

[8] Z. Ning *et al.*, "Partial computation offloading and adaptive task scheduling for 5G-enabled vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 21, no. 4, pp. 1319–1333, Apr. 2022.

[9] Y. Hou *et al.*, "Joint allocation of wireless resource and capability in MEC-enabled vehicular network," *China Commun.*, vol. 18, no. 6, pp. 64–76, Jun. 2021.

[10] H. Guo, J. Liu, and J. Zhang, "Efficient computation offloading for multi-access edge computing in 5G HetNets," in *Proc. IEEE Int. Conf. Commun.*, Kansas City, MO, USA, 2018, pp. 1–6.

[11] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE Int. Symp. Inf. Theory*, Barcelona, Spain, 2016, pp. 1451–1455.

[12] C. M. Huang and C. F. Lai, "The mobile edge computing (MEC)-based vehicle to infrastructure (V2I) data offloading from cellular network to VANET using the delay-constrained computing scheme," in *Proc. IEEE Int. Comput. Symp*, Tainan, Taiwan, 2020, pp. 1–6.

[13] Y. Wu and J. Zheng, "Analysis of the uplink local delay in an MEC-based VANET," in *Proc. IEEE Int. Glob. Commun.*, Waikoloa, HI, USA, 2019, pp. 1–6.

[14] E. T. Michailidis *et al.*, "Energy optimization in massive MIMO UAV-aided MEC-enabled vehicular networks," *IEEE Access*, vol. 9, pp. 117388–117403, Aug. 2021.

[15] S. Zhang *et al.*, "A vehicle-environment cooperative control based velocity profile prediction method and case study in energy management of plug-in hybrid electric vehicles," *IEEE Access*, vol. 7, pp. 75965–75975, Jun. 2019.

[16] J. Fu *et al.*, "Optimization of energy consumption in the MEC-assisted multi-user FD-SWIPT system," *IEEE Access*, vol. 8, pp. 21345–21354, Jan. 2020.

[17] R. Xiong, C. Zhang, X. Yi, L. Li, and H. Zeng, "Joint connection modes, uplink paths and computational tasks assignment for unmanned mining vehicles' energy saving in mobile edge computing networks," *IEEE Access*, vol. 8, pp. 142076–142085, Aug. 2020.

[18] X. Zhang, J. Zhang, J. Xiong, L. Zhou, and J. Wei, "Energy-efficient multi-UAV-enabled multiaccess edge computing incorporating NOMA," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5613–5627, Jun. 2020.

[19] Q. Wu *et al.*, "A task offloading scheme in vehicular fog and cloud computing system," *IEEE Access*, vol. 8, pp. 1173–1184, Dec. 2020.

[20] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint offloading and resource allocation in vehicular edge computing and networks," in *Proc. IEEE Int. Glob. Commun.*, Abu Dhabi, United Arab Emirates, 2018, pp. 1–7.

[21] X. Wang, X. Wei, and L. Wang, "A deep learning based energy-efficient computational offloading method in Internet of Vehicles," *China Commun.*, vol. 16, no. 3, pp. 81–91, Mar. 2019.

[22] A. Rauniyar, P. Engelstad, B. Feng, and D. V. Thanh, "Crowdsourcing-based disaster management using fog computing in Internet of Things paradigm," in *Proc. IEEE Int. Conf. Collaboration. Int. Comput*, Pittsburgh, PA, USA, 2016, pp. 490–494.

[23] W. Zhang, Z. Zhang, and H. Chao, "Cooperative fog computing for dealing with Big Data in the Internet of Vehicles: Architecture and hierarchical resource management," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 60–67, Dec. 2017.

[24] Y. Tan, K. Wang, Y. Yang, and M. Zhou, "Delay-optimal task offloading for dynamic fog networks," in *Proc. IEEE Int. Conf. Commun.*, Shanghai, China, 2019, pp. 1–6.

[25] G. Hu, Y. Jia, and Z. Chen, "Multi-user computation offloading with D2D for mobile edge computing," in *Proc. IEEE Int. Glob. Commun.*, Abu Dhabi, United Arab Emirates, 2018, pp. 1–6.

[26] H. Wang, Z. Lin, K. Guo, and T. Lv, "Energy and delay minimization based on game theory in MEC-assisted vehicular networks," in *Proc. IEEE Int. Conf. Commun. Workshops*, Montreal, QC, Canada, 2021, pp. 1–6.

[27] H. Wang, Z. Lin, K. Guo, and T. Lv, "Computation offloading based on game theory in MEC-assisted V2X networks," in *Proc. IEEE Int. Conf. Commun. Workshops*, Montreal, QC, Canada, 2021, pp. 1–6.

[28] H. Hu, Q. Wang, R. Hu, and H. Zhu, "Mobility-aware offloading and resource allocation in an MEC-enabled IoT network with energy harvesting," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17541–17556, Dec. 2021.

[29] T. D. T. Nguyen, V. Nguyen, V. N. Pham, L. N. T. Huynh, M. D. Hossain, and E. N. Huh, "Modeling data redundancy and cost-aware task allocation in MEC-enabled Internet-of-Vehicles applications," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1687–1701, Feb. 2021.

[30] X. Xu *et al.*, "Service migration across edge devices in 6G-enabled Internet of Vehicles networks," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1930–1937, Feb. 2022.

[31] L. Liang, J. Xiao, Z. Ren, Z. Chen, and Y. Jia, "Particle swarm based service migration scheme in the edge computing environment," *IEEE Access*, vol. 8, pp. 45596–45606, Mar. 2020.

[32] M. R. Anwar, S. Wang, M. F. Akram, S. Raza, and S. Mahmood, "5G enabled MEC: A distributed traffic steering for seamless service migration of Internet of Vehicles," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 648–661, Jan. 2022.

[33] Q. Yuan *et al.*, "A joint service migration and mobility optimization approach for vehicular edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9041–9052, Aug. 2020.

[34] C. Wang *et al.*, "An adaptive deep q-learning service migration decision framework for connected vehicles," in *Proc. IEEE Systems, Man, Cybern.*, Toronto, ON, Canada, 2020, pp. 944–949.

[35] Q. Luo, C. Li, T. H. Luan, and W. Shi, "Collaborative data scheduling for vehicular edge computing via deep reinforcement learning," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9637–9650, Oct. 2020.

[36] W. Xu *et al.*, "Internet of Vehicles in Big Data era," *IEEE/CAA J. Auto. Sinica*, vol. 5, no. 1, pp. 19–35, Jan. 2018.

[37] J. Wang, C. Jiang, Z. Han, Y. Ren, and L. Hanzo, "Internet of Vehicles: Sensing-aided transportation information collection and diffusion," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 3813–3825, May 2018.

[38] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Software-defined networking for RSU clouds in support of the Internet of Vehicles," *IEEE Internet Things J.*, vol. 2, no. 2, pp. 133–144, Apr. 2015.

[39] T. Lian, Y. Zhou, X. Wang, N. Cheng, and N. Lu, "Predictive task migration modeling in software defined vehicular networks," in *Proc. IEEE Int. Conf. Comput. Commun Syst.*, Singapore, 2019, pp. 570–574.

[40] S. C. Ng, W. Zhang, Y. Zhang, Y. Yang, and G. Mao, "Analysis of access and connectivity probabilities in vehicular relay networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 1, pp. 140–150, Jan. 2011.

[41] Y. Wang *et al.*, "Energy-optimal partial computation offloading using dynamic voltage scaling," in *Proc. IEEE Int. Conf. Commun. Workshop.*, London, U.K., 2015, pp. 2695–2700.

[42] J. Zhang *et al.*, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2633–2645, Aug. 2018.

[43] H. W. Kuhn, "Extensive games and the problem of information," in *Contributions to Theory Games, 2 of Annals of Mathematics Studies*, 1953, pp. 193–216.

[44] Y. Liu, S. Wang, J. Huang, and F. Yang, "A computation offloading algorithm based on game theory for vehicular edge networks," in *Proc. IEEE Int. Conf. Commun.*, Kansas City, MO, 2018, pp. 1–6.

[45] H. Zhang, Z. Wang, and K. Liu, "V2X offloading and resource allocation in SDN-assisted MEC-based vehicular networks," *China Commun.*, vol. 17, no. 5, pp. 266–283, May 2020.

[46] Y. Wang, X. Hu, L. Guo, and Z. Yao, "Research on V2I/V2V hybrid multi-hop edge computing offloading algorithm in IoV environment," in *Proc. IEEE Int. Conf. Intell. Transp. Eng.*, Beijing, China, 2020, pp. 336–340.

Haipeng Wang (Student Member, IEEE) received the B.S. degree from the China Jiliang University, Hangzhou, China, in 2018. He is currently working toward the M.S. degree in communication engineering with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, China. His research interests include mobile edge computing, Internet of Vehicles, and task offloading.

Tiejun Lv (Senior Member, IEEE) received the M.S. and Ph.D. degrees in electronic engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 1997 and 2000, respectively. From January 2001 to January 2003, he was a Postdoctoral Fellow with Tsinghua University, Beijing, China. In 2005, he was promoted to a Full Professor with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing. From September 2008 to March 2009, he was a Visiting Professor with the Department of Electrical Engineering, Stanford University, Stanford, CA, USA. He is the author of three books, more than 100 published IEEE journal papers and 200 conference papers on the physical layer of wireless mobile communications. His research interests include signal processing, communications theory, and networking. He was the recipient of the Program for New Century Excellent Talents in University Award from the Ministry of Education, China, in 2006, and the Nature Science Award in the Ministry of Education of China for the hierarchical cooperative communication theory and technologies in 2015.

Zhipeng Lin (Member, IEEE) received the Ph.D. degree from the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing, China, and the School of Electrical and Data Engineering, University of Technology of Sydney, Ultimo, NSW, Australia, in 2021. He is currently an Associate Researcher with the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China. His research interests include signal processing, massive MIMO, spectrum sensing, and UAV communications.

Jie Zeng (Senior Member, IEEE) received the B.S. and M.S. degrees from Tsinghua University, Beijing, China, in 2006 and 2009, respectively, the first Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, in 2019, and the second Ph.D. degree from the University of Technology Sydney, Ultimo, NSW, Australia, in 2021. From July 2009 to May 2020, he was with the Research Institute of Information Technology, Tsinghua University. From May 2020 to April 2022, he was a Postdoctoral Researcher with the Department of Electronic Engineering, Tsinghua University. Since May 2022, he has been an Associate Professor with the School of Cyberspace Science and Technology, Beijing Institute of Technology. He has authored or coauthored more than 100 journal and conference papers, and holds more than 40 Chinese and international patents. His research interests include 5G/6G, URLLC, satellite Internet, and novel network architecture. He participated in drafting one national standard and one communication industry standard in China. He was the recipient of the Beijing's Science and Technology Award of in 2015, Best Cooperation Award of Samsung Electronics in 2016, and Dolby Australia's Best Scientific Paper Award in 2020.