A Game-Based Computation Offloading Method in Vehicular Multiaccess Edge Computing Networks

Yunpeng Wang, Ping Lang, Daxin Tian, *Senior Member, IEEE*, Jianshan Zhou, Xuting Duan, Yue Cao, and Dezong Zhao

***Abstract*—Multiaccess edge computing (MEC) is a new paradigm to meet the requirements for low latency and high reliability of applications in vehicular networking. More computation-intensive and delay-sensitive applications can be realized through computation offloading of vehicles in vehicular MEC networks. However, the resources of a MEC server are not unlimited. Vehicles need to determine their task offloading strategies in real time under a dynamic-network environment to achieve optimal performance. In this article, we propose a multiuser noncooperative computation offloading game to adjust the offloading probability of each vehicle in vehicular MEC networks and design the payoff function considering the distance between the vehicle and MEC access point, application and communication model, and multivehicle competition for MEC resources. Moreover, we construct a distributed best response algorithm based on the computation offloading game model to maximize the utility of each vehicle and demonstrate that the strategy in this algorithm can converge to a unique and stable equilibrium under certain conditions. Furthermore, we conduct a series of experiments and comparisons with other offloading methods to analyze the effectiveness and performance of the proposed algorithms. The fast convergence and the improved performance of this algorithm are verified by numerical results.**

***Index Terms*—Computation offloading, distributed algorithm, game theory, multiaccess edge computing (MEC).**

I. Introduction

WITH the continuous development of intelligent transportation system (ITS) and automated driving technology, a variety of attractive applications have emerged, which cover driving safety, traffic efficiency, information entertainment, and other aspects [1]. A large number of computing-intensive and delay-sensitive applications raise the requirements for computing and storage capacity of vehicles [2]. But limited by the physical space and economic costs, the local resources provided by each vehicle are difficult to fully meet the needs of these applications.

Therefore, multiaccess edge computing (MEC) [3] or formerly mobile-edge computing, as new architecture and key technology for the emerging 5G networks, has been proposed to address this problem [4]. Different from traditional mobile cloud computing (MCC), MEC migrates remote cloud computing resources to the edge of the network to reduce the end-to-end transmission delay of data and to relieve the computing and storage pressure of vehicles or intelligent roadside infrastructures.

However, the computation and storage resources of the MEC server are not unlimited [5]. Continuous increase in the number of vehicles or computing tasks will cause a load of the MEC server to exceed its maximum limit, which makes the MEC unable to guarantee the Quality of Service (QoS) for each vehicle's application, and the vehicle will not benefit from computation offloading in this scenario. Each vehicle needs to consider the requirement for the execution time of its application and the resource occupancy of the MEC server in an integrated manner to decide whether to offloading its task to MEC. Therefore, we can regard the offloading decision of each vehicle as a resource competition problem to control the excessive occupation of MEC resources by users, and to balance the task loads of MEC, so that MEC can serve applications in ITS more efficiently.

In this multivehicle computation offloading scenario, centralized and distributed methods are generally used to control the offloading strategy of each vehicle. The centralized method allocates appropriate MEC resources for each vehicle according to its application requirements through the overall planning of the server side, so as to realize the optimal utilization of resources. In the distributed method, each vehicle perceives the resource occupancy in its environment depending on the interaction between vehicles and decides whether to offload its task to MEC on the premise of maximizing its own utility.

In this article, we focus on the research of the distributed computation offloading method and regard the computation offloading behavior of each vehicle as a competition for resources of a MEC server. Each vehicle can determine its own offloading strategy according to the resource occupation of MEC in the scenario. To this end, we model the interaction among vehicles with a game-theoretic framework and analyze vehicles' offloading strategies under the assumption of rational behavior. The main contributions of this article are as follows.

Manuscript received September 15, 2019; revised December 31, 2019; accepted February 3, 2020. Date of publication February 6, 2020; date of current version June 12, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61672082 and Grant 61822101, and in part by the Beijing Municipal Natural Science Foundation under Grant 4181002. *(Corresponding author: Daxin Tian.)*

Yunpeng Wang, Ping Lang, Daxin Tian, Jianshan Zhou, Xuting Duan, and Yue Cao are with the Beijing Advanced Innovation Center for Big Data and Brain Computing, Beijing Key Laboratory for Cooperative Vehicle Infrastructure Systems and Safety Control, School of Transportation Science and Engineering, Beihang University, Beijing 100191, China (e-mail: ypwang@buaa.edu.cn; lping0852@gmail.com; dtian@buaa.edu.cn; jianshanzhou@foxmail.com; duanxuting@buaa.edu.cn; 871441562@qq.com).

Dezong Zhao is with the Department of Aeronautical and Automotive Engineering, Loughborough University, Loughborough LE11 3TU, U.K. (e-mail: d.zhao@lboro.ac.uk).

Digital Object Identifier 10.1109/JIOT.2020.2972061

1) We propose a multiuser noncooperative computation offloading game of a MEC scenario, in which each vehicle adjusts its offloading probability to achieve the maximum utility.
2) In the computation offloading game, we design the participant's payoff function to evaluate its income, which takes into account the distance between the vehicle and MEC access point, the application and communication model of the vehicle, and the competition for MEC resources of other vehicles.
3) Based on the best response mechanism, we construct a distributed algorithm for the computation offloading game and demonstrate that the strategy in this algorithm can converge to a unique and stable equilibrium.
4) We conduct a series of experiments and comparisons with other offloading methods to analyze the effectiveness and performance of the proposed algorithms.

The remainder of this article is organized as follows. Section II presents related works. In Section III, we describe the basic model of the computation offloading game and design the specific utility function and price function in the payoff function of this game model. Moreover, a distributed algorithm of the computation offloading game is presented in Section V, and the convergence and uniqueness of a Nash equilibrium (NE) are proved therein. Section VI concludes this article and discusses the future work.

II. Related Work

There are many related studies devoted to solving problems of computation offloading of mobile devices in MEC or MCC architectures. Some papers investigated the computation offloading of mobile terminals in single-user scenarios. Aliyu *et al.* [6] proposed a systematic review of MCC energy-aware issues and grouped some research works on battery energy in MCC into dynamic and nondynamic energy-aware task offloading. Abebe and Ryan [7] presented a novel distributed approach to application representation in which each device maintains a graph consisting only of components in its memory space while maintaining abstraction elements for components in remote devices. Their approach removes the need to store and update complete application graphs on each device and reduces the cost of partitioning an application during adaptation. Muñoz *et al.* [8] provided a framework for the joint optimization of the radio and computational resource usage exploiting the tradeoff between energy consumption and latency. Zhang *et al.* [9] provided a theoretical framework of energy-optimal MCC under the stochastic wireless channel to conserve energy for the mobile device, by optimally executing mobile applications in the mobile device or offloading to the cloud. In this framework, they formulated both scheduling problems as constrained optimization problems and obtained closed-form solutions for optimal scheduling policies. To minimize both total tasks' execution latency and the mobile device's energy consumption by jointly optimizing the task allocation decision and the device's central process unit (CPU) frequency, Dinh *et al.* [10] proposed an optimization framework of offloading from a single mobile device to multiple edge devices. Wang *et al.* [11] investigated partial computation offloading by jointly optimizing the computational speed of the smart mobile device, transmit power of the smart mobile device, and offloading ratio with two system design objectives: 1) energy consumption of smart mobile device minimization and 2) latency of application execution minimization.

Some papers focused on the problem of computation offloading in the multiple users' scenario. Chen [12] proposed a game-theoretic approach for achieving efficient computation offloading for MCC and formulated the decentralized computation offloading decision-making problem among mobile device users as a decentralized computation offloading game. Cardellini *et al.* [5] defined a model to capture the user's interaction and to investigate the effects of computation offloading on the users, perceived performance in a three-tier architecture for the mobile computing scenarios consisting of mobile nodes, nearby computing nodes, and distant cloud servers. They formulated the problem as a generalized NE problem and presented a distributed algorithm for the computation of an equilibrium based on an in-depth analysis of the underlying equilibrium problem. Lyu *et al.* [13] proposed a heuristic offloading decision algorithm (HODA), which is semidistributed and jointly optimizes the offloading decision, and communication and computation resources to maximize system utility. Wang *et al.* [14] proposed a unified MEC and wireless power transfer (WPT) design by considering a wireless-powered multiuser MEC system, where a multiantenna access point (integrated with a MEC server) broadcasts wireless power to charge multiple users and each user node relies on the harvested energy to execute computation tasks. Bi and Zhang [15] designed an offloading scheme for the computation rate maximization problem in a multiuser MEC network powered by the WPT, where each energy-harvesting wireless device follows a binary computation offloading policy, i.e., the data set of a task has to be executed as a whole either locally or remotely at the MEC server via task offloading.

In addition, several articles designed communication architectures and computation offloading schemes in vehicular MEC networking. Aliyu *et al.* [16] reviewed cloud computing in vehicular *ad hoc* networks (CC-V) with the emphasis on layered architecture, network component, taxonomy, and future challenges. Specifically, a four-layered architecture for CC-V is proposed, including perception, coordination, artificial intelligence, and smart application layers. Zhou *et al.* [17] investigated dynamic sharing of the 5G spectrum and proposed a sharing architecture of DSRC and the 5G spectrum for immersive experience-driven vehicular communications. In order to make full use of the intelligence at the wireless edge for coordinated content delivery, Yuan *et al.* [18] designed a two-level edge computing architecture for automated driving services and proposed potential solutions to the research challenges of wireless edge caching and vehicular content sharing. Cao *et al.* [19] proposed a MEC-based system enabled by big data analytics for the electric vehicle (EV) charging use case, in which mobility-aware MEC servers interact with opportunistically encountered EVs to disseminate charging stations predicted charging availability, collect EVs driving big data, and implement decentralized computing on data mining and aggregation. Wang *et al.* [2] investigated the vehicular user computation overhead minimization problem in MEC-enabled vehicular networks and developed a low-complexity algorithm by jointly optimizing the computation and communication resources' allocation. Pham *et al.* [4] proposed a scalable vehicle-assisted MEC (SVMEC) paradigm, which can not only relieve the resource limitation of MEC but also enhance the scalability of computing services for IoT devices and reduce the cost of using computing resources. Dai *et al.* [20] proposed integrating load balancing with offloading and studied resource allocation for a multiuser multiserver vehicular edging computing system. Liu *et al.* [21] studied the task offloading problem from a matching perspective and proposed pricing-based matching algorithms to optimize the total network delay. Tan and Hu [22] designed a resource allocation policy of joint communication, caching and computing design problems with deep reinforcement learning and took challenges of the vehicle's mobility and the hard service deadline constraint into account.

Different from the above computation offloading strategies for vehicular MEC networks, we design a multivehicle noncooperative computation offloading game considering the location of the vehicle and the task deadline constraint. On this basis, we propose a lightweight-distributed best response offloading algorithm and ensure that the strategy of each vehicle can converge to the unique NE.

III. System Model

In this section, we first introduce the basic model of the computation offloading game and express the payoff function with each user's utility and overhead. Then, a computation offloading model, including the application model and the communication model, is provided under appropriate assumptions. Finally, we design the specific utility function and price function in the payoff function of this game model. Table I lists all the notations of the used parameters and variables in this section.

TABLE I
Parameters in the System Model

Parameter	Meaning
N	the set of vehicles (i.e., wireless nodes)
n	the number of vehicles
p_i	computation offloading probability of vehicle i
u_i	payoff of vehicle i
$u_i(\mathbf{p})$	payoff function of vehicle i playing the computation offloading game with strategy $\mathbf{p}$
U_i	utility function of vehicle i playing the computation offloading game with strategy $\mathbf{p}$
C_i	price function of vehicle i playing the computation offloading game with strategy $\mathbf{p}$
$\mathbf{p}_{-i}$	computation offloading probability for all vehicles except i
$\mathbf{p}$	computation offloading probability for all vehicles
$\mathbf{p}^*$	Nash equilibrium of computation offloading probability for all vehicles
L_i	input data size of the application in vehicle i
α_i	computational complexity of the application in vehicle i
$t_{i,max}$	completion deadline of the application in vehicle i
$f_{i,l}$	local CPU's computational speed of vehicle i
$t_{i,l}$	local execution time of the application in vehicle i
d_i	distance from vehicle i to MEC access point
θ	path loss exponent
h_i	channel fading coefficient of the communication between vehicle i and MEC access point
N_0	white Gaussian noise power
R_i	data transmission rate between vehicle i and MEC access point
W_i	channel bandwidth of the communication between vehicle i and MEC access point
P_i	transmit power of the communication between vehicle i and MEC access point
$\beta_{i,U}$	overhead in uplink transmission of vehicle i
$\beta_{i,D}$	the combination of downlink transmission overhead and the ratio of output to input bits offloaded from vehicle i to MEC
$t_{i,U}/t_{i,D}$	uplink/downlink transmission delay of vehicle i's computation offloading
$\tau_{i,e}$	execution time of vehicle i's application in MEC
f_e	computational speed of MEC
$t_{i,e}$	total execution time of vehicle i's computation offloading
ρ	pricing factor
λ_j	task arrival rate of vehicle j
$\mathcal{J}$	matrix of slopes of the best-response functions of each vehicle with respect to the strategies of other vehicles

A. Basic Game Model

As depicted in Fig. 1, in ITS and automated driving scenarios, each vehicle will produce some complex computation tasks to meet the safety, entertainment, and other needs of its driver and passengers. Because of the limited computing capacity of the vehicle, many computation tasks are difficult to be completed locally. The vehicle can offload the task to the roadside MEC platform in order to achieve higher computing efficiency. However, as stated in the Introduction, the computation and storage capacity of the MEC server is not unlimited. It needs to charge different fees to vehicles according to its own resource occupation to ensure the stability of its load. Therefore, a vehicle can balance its own utility and cost through a game to determine whether the computation task should be offloading to the MEC platform to maximize its own benefits.

In this article, we consider a set $N = \{1, 2, \ldots, n\}$ of vehicles in the coverage of a MEC access point. For simplicity, we only study the computation offloading game under a single MEC platform, i.e., we do not consider the handover problem between different MEC platforms. We assume each vehicle can acquire the offloading probability of the other vehicles in the previous stage from MEC in the game.

Now, we can define the basic computation offloading game as a triple $\mathcal{G} = \{N, (p_i)_{i\in N}, (u_i)_{i\in N}\}$, where $N = \{1, 2, \ldots, n\}$ is the set of players (vehicles), $p_i \in [0, 1]$ is the offloading probability of vehicle i or the mixed strategy of the ith vehicle in the game, and u_i is the payoff acquired from payoff function $u_i(\mathbf{p})$ of vehicle i.

1) *Players:* We consider set N of vehicles in the coverage of a MEC access point, $N = \{1, 2, \ldots, n\}$, $i \in N$.

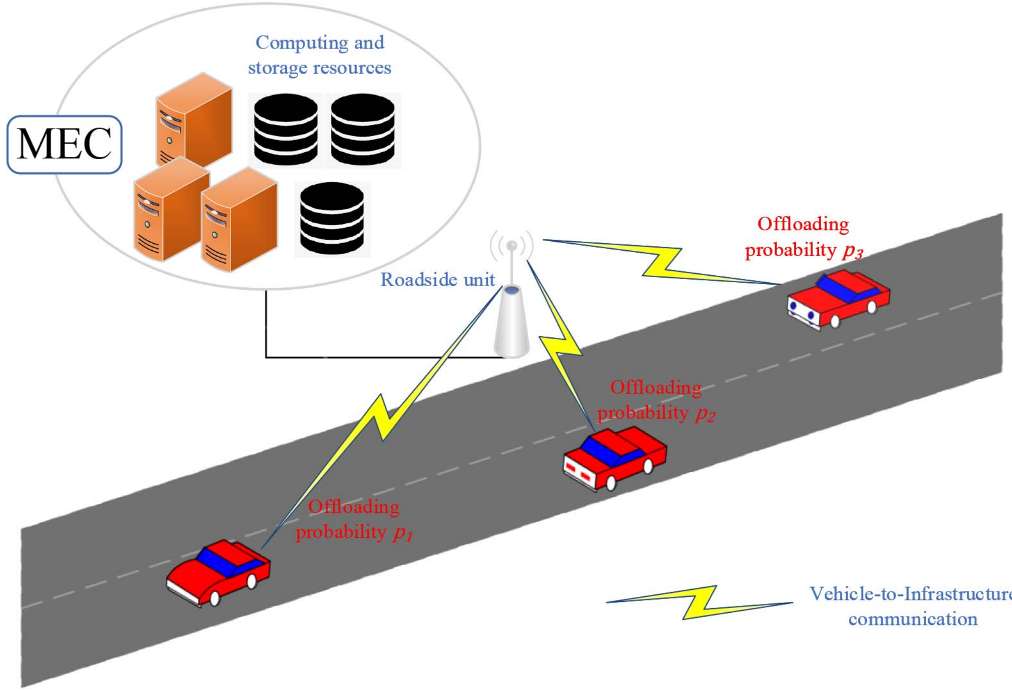

Fig. 1. Computation offloading in vehicular MEC networks.

2) *Strategy:* Action set of a vehicle $A_i = \{E, L\}$, where actions E and L represent the MEC execution and local execution of vehicle i's task, respectively, i.e., whether vehicle i offloads its task to the MEC for execution. Pure strategy s_i is the mapping of the current state to vehicle's action set. Vehicle's mixed strategy p_i is the probability distribution of the pure strategy at the current stage of the node. The control actions of vehicle i's strategy can be defined as follows:

$$p_i = \phi_i(p_{1\,:\,n}) = \arg\max_{p_i} u_i(\mathbf{p}). \tag{1}$$

In the next section, we will provide a specific strategy update mechanism based on the best response method.

3) *Payoff Function:* The payoff function of vehicle i is expressed as $u_i(\mathbf{p})$, where the node maximizes the global payoff of the game. The basic framework is defined as

$$u_i(\mathbf{p}) = U_i(\mathbf{p}) - C_i(\mathbf{p}) \tag{2}$$

where $U_i(\mathbf{p})$ and $C_i(\mathbf{p})$ represent the utility function and the price function of vehicle i, respectively. The exact form of this payoff function will be described later.

Let $\mathbf{p}_{-i} = (p_1, \ldots, p_{i-1}, p_{i+1}, \ldots, p_n)$ denote the offloading probability for all vehicles except i, then we can introduce the following concept of NE in the computation offloading game [23].

Definition 1: A computation offloading probability vector $\mathbf{p}^*$ is said to be an NE if no vehicle can improve its payoff by unilaterally deviating from the NE, i.e.,

$$u_i\big(p_i^*, \mathbf{p}_{-i}^*\big) \geq u_i\big(p_i, \mathbf{p}_{-i}^*\big) \quad \forall p_i. \tag{3}$$

In a computation offloading game, each vehicle will consider the latency benefit and communication overhead of task offloading comprehensively in its payoff function. Therefore, in the next section, we introduce an offloading model to analyze the latency to execute a task in the local computing and offloading scenarios.

B. Computation Offloading Model

In the computation offloading game, a vehicle needs an explicit computation offloading model to determine the benefits and costs of task offloading. Therefore, in this section, we introduce the specific application and communication model and analyze the latency of task executed in both local and offloading situations.

Following [9] and [11], we can use three parameters to abstract and depict the application in vehicle i as $(L_i, \alpha_i, t_{i,\max})$, where L_i, α_i, and $t_{i,\max}$ denote the input data size, computational complexity, and completion deadline of application, respectively. The input data size and computational complexity of an application will affect the execution time of the task, both locally and MEC. The L_i-bit input data will also affect the transmission time of tasks in the case of computation offloading. If a task can be completed ahead of deadline $t_{i,\max}$, the vehicle can get a higher utility, otherwise, it will produce the corresponding loss. In addition, the task arrival process is assumed as a Poisson process with an average rate λ_i in vehicle i. For simplicity, we regard the application of each vehicle as a unit task, i.e., only consider the full application offloading in this article.

Then, with vehicle i's local CPU's computational speed $f_{i,l}$, the local execution time is given by

$$t_{i,l} = \frac{\alpha_i L_i}{f_{i,l}}. \tag{4}$$

In this article, we assume the communication model between the vehicle and the MEC access point as a frequency-flat block-fading Rayleigh channel, with block length no less than the completion deadline of application [11]. We model the path loss as $d_i^{-\theta}$, where d_i and θ denote the distance from vehicle i to MEC access point and the path loss exponent, respectively. In addition, the channel fading coefficient and the white Gaussian noise power can be denoted as h_i and N_0, respectively.

Therefore, the data transmission rate between vehicle i and the MEC access point is expressed as

$$R_i = W_i \log_2\left(1 + \frac{P_i d_i^{-\theta}|h_i|^2}{N_0}\right) \tag{5}$$

where W_i and P_i denote the channel bandwidth and transmit power of the communication between vehicle i and the MEC access point.

Following [11], we use $\beta_{i,U}$ as the overhead of uplink and $\beta_{i,D}$ as the combination of downlink transmission overhead and the ratio of output to input bits offloaded. Then, the uplink and downlink transmission delays of vehicle i's computation offloading are given by

$$t_{i,U} = \frac{\beta_{i,U} L_i}{R_i} \tag{6}$$

$$t_{i,D} = \frac{\beta_{i,D} L_i}{R_i}. \tag{7}$$

The execution time of vehicle i's application in MEC is given by

$$\tau_{i,e} = \frac{\alpha_i L_i}{f_e} \tag{8}$$

where f_e is the computational speed of MEC. So, the total execution time of vehicle i's computation offloading can be derived as

$$\begin{aligned} t_{i,e} &= t_{i,U} + \tau_{i,e} + t_{i,D} \\ &= \frac{\beta_{i,U} L_i}{W_i \log_2\left(1 + \frac{P_i d_i^{-\theta}|h_i|^2}{N_0}\right)} + \frac{\alpha_i L_i}{f_e} \\ &\quad + \frac{\beta_{i,D} L_i}{W_i \log_2\left(1 + \frac{P_i d_i^{-\theta}|h_i|^2}{N_0}\right)}. \end{aligned} \tag{9}$$

Now, based on the computation offloading model, we obtain the time when the task of vehicle i is executed locally and offloaded to the MEC server, respectively. Next, we will use these results to design the utility function and price function of the vehicle and to reflect the real benefits of the vehicle in the computation offloading game.

C. Payoff Function

Taking into account $t_{i,l}$, $t_{i,e}$, and $t_{i,\max}$ analyzed in the previous section, we can use the execution time of the task as the basis for evaluating the utility of vehicle. In other words, when an application is executed locally or on a MEC server, the shorter the execution time of the task, the more utility the vehicle can achieve. On the contrary, if the execution time of the task exceeds the maximum execution time $t_{i,\max}$ which the vehicle can tolerate, the vehicle will not be able to obtain benefit and be punished accordingly.

Based on this principle, the vehicle i's utility function in a computation offloading game is derived as

$$U_i(\mathbf{p}) = \frac{t_{i,\max} - t_{i,l}}{t_{i,\max}}(1 - p_i) + \frac{t_{i,\max} - t_{i,e}}{t_{i,\max}} p_i \tag{10}$$

where $[(t_{i,\max} - t_{i,l})/t_{i,\max}]$ denotes the utility of task executed locally and $[(t_{i,\max} - t_{i,e})/t_{i,\max}]$ denotes the computation offloading utility of vehicle i's task. If the execution time of task exceeds $t_{i,\max}$, the vehicle will obtain a negative utility in (10) to show the penalty for its strategy.

As mentioned at the beginning of this section, the computing and storage resources of the MEC server are also limited, so a dynamic pricing mechanism is needed to control the offloading behavior of vehicles. That is, when too many vehicles offload their tasks to the MEC server, the cost (i.e., price) of computation offloading will be raised significantly. Conversely, when the resources of the MEC server are idle, vehicles can offload their tasks to the MEC server at a lower price. This mechanism ensures that a load of a MEC platform does not exceed its maximum limit and maintains the whole system in a relatively balanced state.

The price function with a dynamic pricing mechanism is expressed as

$$C_i(\mathbf{p}) = p_i^2 \rho \left[1 - \prod_{j \neq i}\left(1 - \lambda_j p_j\right)\right] \tag{11}$$

where $\rho \in [0, 1]$ is the pricing factor to adjust prices externally and λ_j denotes the task arrival rate of vehicle j. Using the pricing factor, the operator of MEC servers can adjust the intensity of the resource competition between vehicles in the system artificially with the number of deployed server resources. If there are more computing and storage resources in the MEC server, a smaller pricing factor can be set so that each vehicle has a stronger willingness to offload its task to the MEC server, so as to make full use of MEC resources.

In addition, we can find in (11) that when the pricing factor is fixed, the increase in the number of offloaded vehicles will raise the computation offloading cost of vehicles effectively. Similarly, the increase in computation offloading probabilities and task arrival rates of vehicles will also aggravate the resource occupation of the MEC server, which will lead to an increase in the offloading cost of vehicles in (11) to weaken the willingness of each vehicle's computation offloading. Therefore, the price function in (11) successfully portrays the computation offloading behavior of each vehicle as a competition for server resources, which ensures the load balance of the MEC server and meets the purpose of designing this function.

Finally, by substituting (10) for utility function $U_i(\mathbf{p})$ and (11) for pricing function $C_i(\mathbf{p})$ in payoff function (2), the complete payoff function of vehicle i in the computation offloading game is expressed as

$$u_i(\mathbf{p}) = \frac{t_{i,\max} - t_{i,l}}{t_{i,\max}}(1 - p_i) + \frac{t_{i,\max} - t_{i,e}}{t_{i,\max}} p_i - p_i^2 \rho \left[1 - \prod_{j \neq i} \big(1 - \lambda_j p_j\big) \right]. \tag{12}$$

In the next section, we discuss a specific algorithm according to the computation offloading game model and prove the convergence and uniqueness of an NE.

IV. Distributed Solution

In the previous games theoretical analysis, we addressed a computation offloading game among vehicles to help the vehicle decide whether to offload its computation task to the MEC server for execution. As stated in the existence theorem of the NE in [24], there must exist a mixed strategy NE in this offloading game. However, we still need to find a mechanism to make the participants' mixed strategy converge to the equilibrium and analyze whether the equilibrium is unique. Therefore, in the following discussion, we study the best response strategy of each participant, design the corresponding distributed best response algorithm for computation offloading game, and prove the strategy update mechanism in this algorithm can guarantee the convergence and uniqueness of the NE under certain conditions. According to the strategy of vehicles other than vehicle i, the best response strategy update of player i is expressed as

$$p_i = \arg\max_{p_i} u_i(\mathbf{p}) = \left[\frac{t_{i,l} - t_{i,e}}{2\rho t_{i,\max}\Big(1 - \prod_{j \neq i}\big(1 - \lambda_j p_j\big)\Big)} \right]_0^1 \tag{13}$$

where the operator $[x]_0^1$ can cause $p_i \in [0, 1]$ [25]. Considering the fact that the only information available for any vehicle i at the stage s is the previous offloading probability of other vehicles, offloading probabilities of other vehicles will be represented as p_j^{s-1}. Then, the distributed best response algorithm for computation offloading game is defined as Algorithm 1.

Algorithm 1 Distributed Best Response Algorithm for Computation Offloading Game

1: Initialization: stage $s = 0$, the offloading probability vector $\mathbf{p}^s$, the pricing vector $\boldsymbol{\rho}$, and the task arrival rate vector $\boldsymbol{\lambda}$.
2: Locally at each node i, iterate through s
3: Set $s \leftarrow s + 1$.
4: **for all** $i \in \{1, \ldots, n\}$ **do**
5: Calculate the local execution time of the task without computation offloading,
$$t_{i,l} = \frac{\alpha_i L_i}{f_{i,l}}.$$
6: Estimate the data transmission rate based on distance between vehicle and MEC access point,
$$R_i = W_i \log_2\left(1 + \frac{P_i d_i^{-\theta} |h_i|^2}{N_0}\right).$$
7: Calculate the total execution time of the task with offloading,
$$t_{i,e} = t_{i,U} + \tau_{i,e} + t_{i,D} = \frac{\beta_{i,U} L_i}{R_i} + \frac{\alpha_i L_i}{f_e} + \frac{\beta_{i,D} L_i}{R_i}.$$
8: Update the best-response strategy according to offloading probabilities of other vehicles in the previous stage,
$$p_i^s = \left[\frac{t_{i,l} - t_{i,e}}{2\rho t_{i,max}\Big(1 - \prod_{j \neq i}\Big(1 - \lambda_j p_j^{s-1}\Big)\Big)} \right]_0^1.$$
9: **if** p_i^s has converged **then**
10: Vehicle i decides whether to offload its task with a probability p_i^s.
11: **end if**
12: **end for**
13: Go to Step 3.

Following [25]–[27], we can prove that the best response strategy of a computation offloading game in Algorithm 1 converges to a unique NE by showing (13) is a contraction mapping.

Lemma 1: If the best response mapping is a contraction on the entire strategy space, there is a unique NE in the game.

Theorem 1: Regardless of any initial vector $\mathbf{p}^0$, the iteration defined by best response in Algorithm 1 can converge to the unique NE of the computation offloading game if

$$\sum_{j \neq i} \frac{\big|t_{i,l} - t_{i,e}\big| \prod_{k \neq i,j}(1 - \lambda_k p_k)}{2\rho t_{i,\max}\Big(1 - \prod_{j \neq i}\big(1 - \lambda_j p_j\big)\Big)^2} < 1 \quad \forall i \in \{1, \ldots, n\}. \tag{14}$$

Proof: According to Lemma 1 and the contraction mapping theorem in [28], we only need to show the updating rule in (13) is a contraction mapping and derive conditions by verifying the infinite norm of its Jacobian matrix $\mathcal{J}$ is less than one.

First, we construct the Jacobian matrix $\mathcal{J}_{n\times n} := (J_{i,j})$ of the best response dynamic of (13) as

$$J_{i,j} = \frac{\partial p_i^s}{\partial p_j^{s-1}} = \begin{cases} 0, & i = j \\ \dfrac{-(t_{i,l} - t_{i,e}) \prod_{k \neq i,j}(1 - \lambda_k p_k)}{2\rho t_{i,\max}\Big(1 - \prod_{j \neq i}\big(1 - \lambda_j p_j\big)\Big)^2}, & \text{otherwise.} \end{cases} \tag{15}$$

That is, $\mathcal{J}$ is the matrix of slopes of the best response functions in Algorithm 1 of each vehicle with respect to the strategies of other vehicles. Then, the infinite norm of it is written as

$$\|\mathcal{J}\|_\infty = \max_{i \in \{1,\ldots,n\}} \sum_{j \neq i} \frac{|t_{i,l} - t_{i,e}| \prod_{k \neq i,j}(1 - \lambda_k p_k)}{2\rho t_{i,\max}\Big(1 - \prod_{j \neq i}\big(1 - \lambda_j p_j\big)\Big)^2}. \tag{16}$$

$\|\mathcal{J}\|_\infty$ in (16) denotes the maximum value of the sum of absolute values of the off-diagonal elements in $\mathcal{J}$.

TABLE II
PARAMETERS IN NUMERICAL ANALYSIS [2], [8]

Parameter	Meaning	Value
L	Task input data size	1 Mbits
α	Task computational complexity	240 cycles/bit
f_l	Local computation capability	1 GHz
f_e	MEC computation capability	5 GHz
θ	Path loss exponent	2
W	Channel bandwidth	10 MHz
P	Transmit power	0.2 W
$\beta_{i,U}$	Overhead of uplink transmission	1
$\beta_{i,D}$	Combination of downlink transmission overhead and the ratio of output to input bits offloaded	0.05
λ	Task arrival rate	0.7
ρ	MEC pricing factor	0.7

Therefore, if $\forall i \in \{1, \ldots, n\}$, $\sum_{j \neq i}([|t_{i,l} - t_{i,e}| \prod_{k \neq i,j}(1 - \lambda_k p_k)]/[2\rho t_{i,\max}(1 - \prod_{j \neq i}(1 - \lambda_j p_j))^2]) < 1$, $\|\mathcal{J}\|_\infty$ will be less than one and (13) will become a contraction mapping. The convergence and uniqueness of p_i in Algorithm 1 are thus guaranteed. ■

In general, considering that the computing and communication capability of the MEC server is much larger than that of a vehicle, the offloading probability in Algorithm 1 can converge to a unique NE in most cases, which is also illustrated by the experimental results in the next section.

V. NUMERICAL RESULTS

In this section, we run some numerical experiments and plot analytical results in MATLAB to verify the convergence of offloading probability in Algorithm 1 and to analyze the variation of vehicle offloading probability with different parameters in the model. In addition, we compare the performance (e.g., expect payoff and latency) of the distributed best response algorithm for computation offloading game with other computation offloading solutions.

For simplicity without losing generality, we consider that the communication range of a MEC access point is 200 m and vehicles are independently and uniformly distributed in its coverage. Moreover, all vehicles share the same input data size, computational complexity, task arrival rate, and some other parameters, i.e., for each vehicle i $L_i = L$, $\alpha_i = \alpha$, $f_{i,l} = f_l$, $\lambda_i = \lambda$, $W_i = W$, and $P_i = P$. Table II lists the default parameter setting in numerical experiments.

Fig. 2 verifies the convergence of each vehicle's computation offloading probability in Algorithm 1 with six vehicles in the coverage range of a MEC access point. It is known that if the best response dynamics reach a steady state, then this state is an equilibrium. In Fig. 2, we can see that the offloading probability of each vehicle converges to a stable NE rapidly within 15 iterations in Algorithm 1. This verifies that the best response of vehicles in the algorithms can converge to unique NE if $\forall i \in \{1, \ldots, n\}$, $\sum_{j \neq i}([|t_{i,l} - t_{i,e}| \prod_{k \neq i,j}(1 - \lambda_k p_k)]/[2\rho t_{i,\max}(1 - \prod_{j \neq i}(1 - \lambda_j p_j))^2]) < 1$. Moreover, in this computation offloading game, due to the different distances between MEC access point and vehicles, offloading probabilities of vehicles under different path losses and channel fading are quite different, which will be further analyzed later.

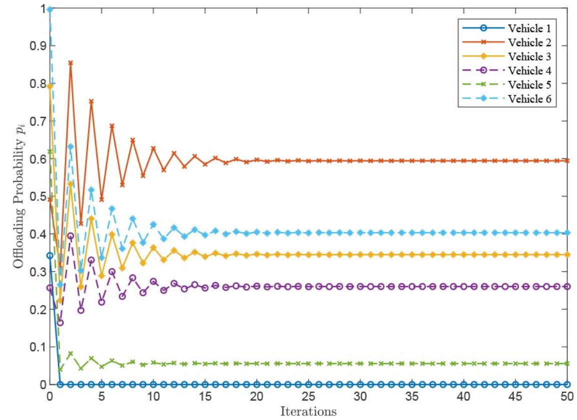

Fig. 2. Convergence process of each vehicle's offloading probability with $n = 6$.

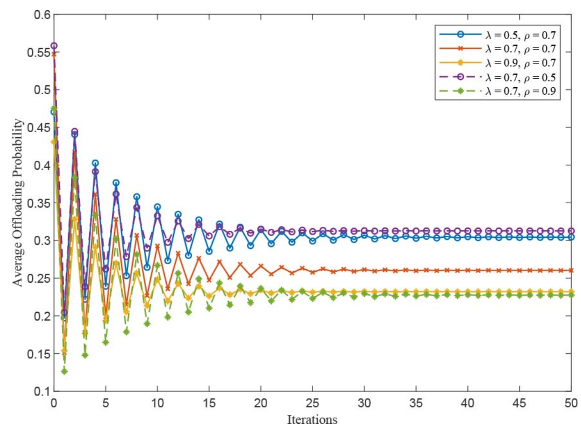

Fig. 3. Convergence process of average offloading probability under different λ and ρ with $n = 10$.

Fig. 3 shows the convergence process of average offloading probability under different task arrival rates and pricing factors with ten vehicles in the game. Similar to Fig. 2, the average offloading probability converges to a stable equilibrium under different conditions. It can be observed that with the increase of task arrival rate, the equilibrium of the average offloading probability decreases gradually. This is mainly because the increase of tasks leads to the high load of the MEC server, and the offloading cost of each participant also increases, which makes each vehicle reduce the offloading probability to ensure its own utility. Similarly, the higher pricing factor also leads to the reduction of the average offloading probability of vehicles in the game.

Fig. 4 displays the variation in the vehicle's offloading probability as a function of the distance between the MEC access point and a vehicle within the communication range under different task arrival rates and pricing factors with ten vehicles in the game. It can be observed that with the increase in the distance d_i, vehicle i's computation offloading probability decreases gradually. This is because the increase of distance increases the path loss of communication, which increases the

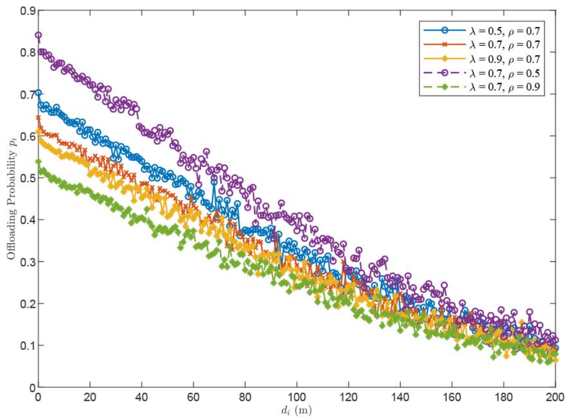

Fig. 4. Vehicle's offloading probability versus d_i under different λ and ρ with $n = 10$.

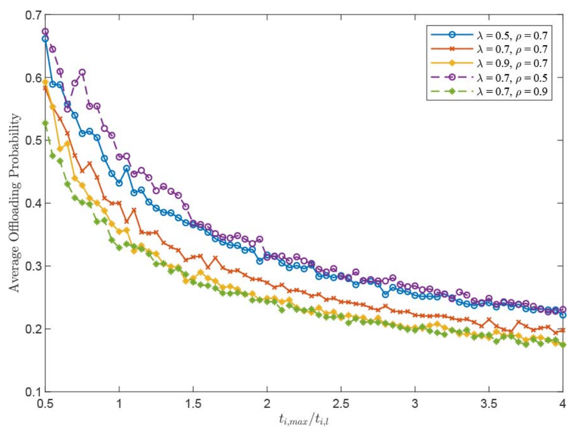

Fig. 5. Average offloading probability versus ratio of $t_{i,\max}$ to $t_{i,l}$ under different λ and ρ with $n = 10$.

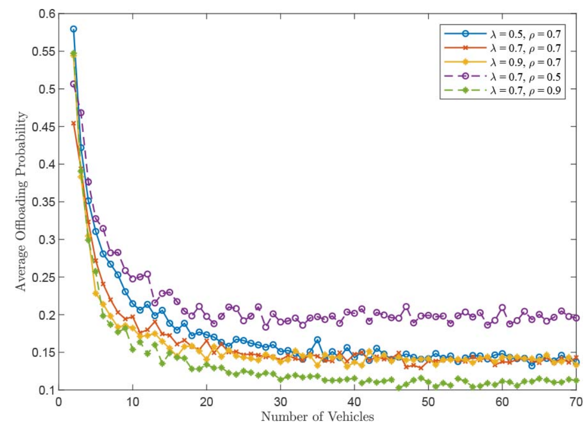

Fig. 6. Average offloading probability versus number of vehicles under different λ and ρ.

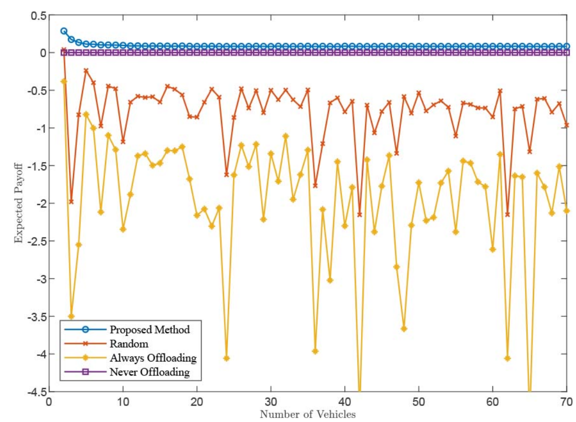

Fig. 7. Expected payoff versus number of vehicles for different offloading methods.

execution delay of tasks in MEC and reduces the willingness of the vehicle for computation offloading. In addition, the fluctuation of offloading probability in the figure is caused by the fading of the Rayleigh channel in the communication model.

Fig. 5 depicts the variation in average offloading probability as a function of the ratio of task execution deadline to its local execution time under different task arrival rates and pricing factors with ten vehicles in the game. As we can see, with the increase of the ratio of $t_{i,\max}$ to $t_{i,l}$, the average offloading probability of vehicles decreases gradually and tends to be stable. When the ratio is less than 1, the vehicle cannot execute the task in time locally, and it can only obtain utility by computation offloading. When the ratio becomes larger, the vehicle can execute the task itself, and its willingness for offloading becomes lower, resulting in a lower average offloading probability.

Fig. 6 shows the variation in average offloading probability as a function of the number of vehicles within the coverage range of a MEC access point under different task arrival rates and pricing factors in the game. As can be seen from Fig. 6, the number of vehicles has a significant impact on the average offloading probability. With the increase in the number of vehicles, the average offloading probability of vehicles decreases sharply, and it finally stabilizes at a lower value. The increase in the number of vehicles in the game makes the load of MEC increase rapidly and reach a saturated state. At the same time, the offloading overhead of each vehicle also increases significantly, resulting in a lower offloading probability for a vehicle to ensure its own payoff. In addition, when a load of the MEC server has reached saturation, the pricing mechanism within the game can no longer make sense, so task arrival rates cannot affect average offloading probability obviously. At this time, the external pricing factor ρ can play a better role in the adjustment of the price function.

Fig. 7 displays the comparison of expected payoff under different computation offloading schemes with $t_{i,\max} = t_{i,l}$. Through the continuous game between vehicles in the proposed method of this article, each vehicle can maintain the expected payoff at a higher level. Since the deadline for the task execution in this experiment is the same as the local execution time, the payoff obtained from the scheme of never offloading is constant to 0. However, random and always offloading schemes cannot ensure the vehicle gets a higher payoff because their strategies make the vehicle pay the cost but cannot get the higher utility by computation unloading. As a result, the expected payoff value of each vehicle is low and fluctuating.

Fig. 8 shows the comparison of expected latency under different computation offloading schemes with $t_{i,\max} = t_{i,l}$.

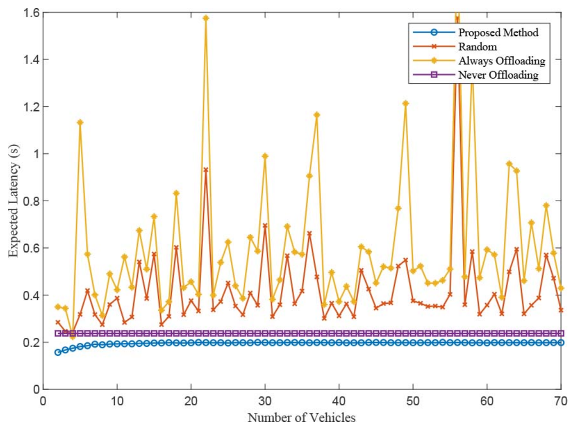

Fig. 8. Expected latency versus number of vehicles for different offloading methods.

Similar to the case in Fig. 7, the local execution time of the task in this experiment is the same as its deadline, so the expected latency from never offloading scheme can be used as a criterion to determine whether different offloading schemes can meet the requirement of task execution latency. By comparison, we can see that the expected latency from the game method will not exceed the deadline of the task and sustain at a low value under different vehicle numbers. However, due to the different distances between vehicle and MEC access point, which lead to different path losses and channel fading, random and never offloading schemes cannot maintain the expected latency at a stable level. Therefore, this experiment embodies the advantages of the computation offloading game proposed in this article in controlling task execution latency.

VI. Conclusion

In this article, we regard the computation offloading behavior of each vehicle as a competition for resources of a server in vehicular MEC networks and proposed a multiuser noncooperative computation offloading game of a MEC scenario, in which each vehicle adjusts its offloading probability to achieve the maximum utility. The payoff function in this game was designed with considering the distance between the vehicle and MEC access point, application and communication model, and multivehicle competition for MEC resources. We analyzed the dynamics of vehicle interactions and constructed a distributed best response algorithm for the computation offloading game. Furthermore, we proved that the offloading probability of each vehicle in this algorithm can converge to a unique equilibrium under certain conditions.

Through a series of experiments, we verified the convergence of the vehicle's offloading probability in the best response algorithms and analyzed the influence of parameters in the computation offloading game on the performance of the distributed algorithm. Moreover, we verified the performance improvement of the proposed algorithm by comparing the expected payoff and latency of the game method with other computation offloading solutions. In the future, we will extend this game model to multivehicle and multi-MEC collaborative environments where vehicles can choose to offload their tasks to multiple MECs or other computational vehicles nearby to maximize their utility.

References

[1] J. A. Guerrero-Ibanez, S. Zeadally, and J. Contreras-Castillo, "Integration challenges of intelligent transportation systems with connected vehicle, cloud computing, and Internet of Things technologies," *IEEE Wireless Commun.*, vol. 22, no. 6, pp. 122–128, Dec. 2015.

[2] J. Wang, D. Feng, S. Zhang, J. Tang, and T. Q. Quek, "Computation offloading for mobile edge computing enabled vehicular networks," *IEEE Access*, vol. 7, pp. 62624–62632, 2019.

[3] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[4] 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.

[5] V. Cardellini *et al.*, "A game-theoretic approach to computation offloading in mobile cloud computing," *Math. Program.*, vol. 157, no. 2, pp. 421–449, 2016.

[6] A. Aliyu, A. H. Abdullah, O. Kaiwartya, M. J. Usman, and S. O. A. Rahman, "Mobile cloud computing energy-aware task offloading (MCC: ETO)," in *Proc. Int. Conf. Commun. Comput. Syst.(ICCCS)*, 2017, p. 359.

[7] E. Abebe and C. Ryan, "Adaptive application offloading using distributed abstract class graphs in mobile environments," *J. Syst. Softw.*, vol. 85, no. 12, pp. 2755–2769, 2012.

[8] O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Trans. Veh. Technol.*, vol. 64, no. 10, pp. 4738–4755, Oct. 2015.

[9] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sep. 2013.

[10] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Trans. Commun.*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017.

[11] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

[12] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[13] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.

[14] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[15] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.

[16] A. Aliyu *et al.*, "Cloud computing in vanets: Architecture, taxonomy, and challenges," *IETE Tech. Rev.*, vol. 35, no. 5, pp. 523–547, 2018.

[17] H. Zhou, W. Xu, Y. Bi, J. Chen, Q. Yu, and X. S. Shen, "Toward 5G spectrum sharing for immersive-experience-driven vehicular communications," *IEEE Wireless Commun.*, vol. 24, no. 6, pp. 30–37, Dec. 2017.

[18] Q. Yuan, H. Zhou, J. Li, Z. Liu, F. Yang, and X. S. Shen, "Toward efficient content delivery for automated driving services: An edge computing solution," *IEEE Netw.*, vol. 32, no. 1, pp. 80–86, Jan./Feb. 2018.

[19] Y. Cao *et al.*, "Mobile edge computing for big-data-enabled electric vehicle charging," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 150–156, Mar. 2018.

[20] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint load balancing and offloading in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4377–4387, Jun. 2019.

[21] P. Liu, J. Li, and Z. Sun, "Matching-based task offloading for vehicular edge computing," *IEEE Access*, vol. 7, pp. 27628–27640, 2019.

[22] L. T. Tan and R. Q. Hu, "Mobility-aware edge caching and computing in vehicle networks: A deep reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10190–10203, Nov. 2018.

[23] S. Tadelis, "Game theory: An introduction," in *Economics Books*, vol. 1. Princeton, NJ, USA: Princeton Univ. Press, 2012.

[24] D. Fudenberg and J. Tirole, *Game Theory*. Cambridge, MA, USA: MIT Press, 1991.

[25] P. Lang, J. Wang, F. Mei, and W. Deng, "A vehicle's weight-based prioritized reciprocity MAC," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 12, 2019, Art. no. e3654.

[26] G. P. Cachon and S. Netessine, "Game theory in supply chain analysis," *Models, Methods, and Applications for Innovative Decision Making*. Catonsville, MD, USA: INFORMS, 2006, pp. 200–233.

[27] J.-W. Lee, A. Tang, J. Huang, M. Chiang, and A. R. Calderbank, "Reverse-engineering MAC: A non-cooperative game model," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 6, pp. 1135–1147, Aug. 2007.

[28] R. Abraham, J. E. Marsden, and T. S. Ratiu, *Manifolds, Tensor Analysis, and Applications*. New York, NY, USA: Springer-Verlag, 1988.

Yunpeng Wang received the Ph.D. degree in mechanical engineering from the Jilin University of Technology, Changchun, China.

He is currently a Professor with the School of Transportation Science and Engineering, Beihang University, Beijing, China. His current research interests include intelligent transportation systems and vehicle infrastructure integration.

Ping Lang received the B.Eng. and M.Sc. degrees in computer science from Jilin University, Changchun, China, in 2016 and 2019, respectively. He is currently pursuing the Ph.D. degree with the School of Transportation Science and Engineering, Beihang University, Beijing, China.

His research interests include vehicular networks and intelligent transportation systems.

Daxin Tian (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science from Jilin University, Changchun, China, in 2002, 2005, and 2007, respectively.

He is currently a Professor with the School of Transportation Science and Engineering, Beihang University, Beijing, China. His current research interests include mobile computing, intelligent transportation systems, vehicular *ad hoc* networks, and swarm intelligent.

He is an IEEE Intelligent Transportation Systems Society Member and an IEEE Vehicular Technology Society Member.

Jianshan Zhou received the B.Sc. and M.Sc. degrees in traffic information engineering and control from Beihang University, Beijing, China, in 2013 and 2016, respectively, where he is currently pursuing the Ph.D. degree with the School of Transportation Science and Engineering.

His current research interests include wireless communication, artificial intelligent system, and intelligent transportation systems.

Xuting Duan received the Ph.D. degree in traffic information engineering and control from Beihang University, Beijing, China.

He is currently an Assistant Professor with the School of Transportation Science and Engineering, Beihang University. His current research interests include vehicular *ad hoc* networks, cooperative vehicle infrastructure system, and Internet of Vehicles.

Yue Cao received the Ph.D. degree from the University of Northumbria, Newcastle upon Tyne, U.K.

He has been a Senior Lecturer with the University of Northumbria and an International Lecturer with Lancaster University, Lancashire, U.K. He is currently a Professor with the School of Transportation Science and Engineering, Beihang University, Beijing, China. His current research interests include connected and automated vehicles, information security in CVIS, and traffic planning and charge management in new energy vehicles.

Dezong Zhao received the B.Eng. and M.Sc. degrees in control science and engineering from Shandong University, Jinan, China, in 2003 and 2006, respectively, and the Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 2010.

He is currently an Assistant Professor with the Department of Aeronautical and Automotive Engineering, Loughborough University, Loughborough, U.K. His current research interests include connected and automated vehicles, autonomous vehicles, machine learning, and dynamic optimization.