Vehicular Task Offloading via Heat-Aware MEC Cooperation Using Game-Theoretic Method

Zhu Xiao, *Senior Member, IEEE*, Xingxia Dai, Hongbo Jiang, *Senior Member, IEEE*, Dong Wang, *Member, IEEE*, Hongyang Chen, *Senior Member, IEEE*, Liang Yang, *Senior Member, IEEE*, and Fanzi Zeng, *Member, IEEE*

***Abstract*—Mobile-edge computing (MEC) has been witnessed as a promising solution for the vehicular task offloading. Due to the limited computing resource of individual MEC servers, it faces challenges when higher requirements are put forward for timely task processing of a large amount of computations in the emerging vehicular applications. In this article, we strive to realize the efficient vehicular task offloading via heat-aware MEC cooperation from the game theory perspective. Here, the heat indicates the vehicle density and is tightly related to the requests of vehicle users when they drive through the hot zones. Specifically, a deep learning-based prediction method is proposed, capturing the dynamic time-varying heat value of the hot zones based on the analysis of the real-world private car trajectory data. To identify the role of MEC in the cooperation, we take the time-delay constraint into consideration for the task offloading. To realize MEC grouping for task offloading in MEC cooperation, we formulate the MEC grouping as a utility maximization problem via designing a noncooperative game-theoretic strategy selection based on regret-matching. Furthermore, we derive the correlated equilibrium and prove that the fast convergence can be achieved. Extensive simulation results validate the effectiveness of the proposed vehicular task offloading approach under various system parameters, such as computation workload, time slots, and MEC servers number. The proposed method outperforms the existing methods, which is able to significantly reduce the task complete delay, and in the meantime enhance the MEC energy efficiency with end users' quality-of-experience guaranteed.**

***Index Terms*—Correlated equilibrium, mobile-edge computing (MEC), private cars, vehicular task offloading.**

Manuscript received November 10, 2019; revised December 7, 2019 and December 11, 2019; accepted December 15, 2019. Date of publication December 18, 2019; date of current version March 12, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61772184, Grant 61732017, and Grant 61572219; in part by the Key Research and Development Project of Hunan Province of China under Grant 2018GK2014; in part by the Open Fund of State Key Laboratory of Geo-Information Engineering under Grant SKLGIE2018-M-4-3; and in part by the Scientific Research Project of Department of the Natural Resources of Hunan Province under Grant 201910. *(Corresponding authors: Hongbo Jiang; Fanzi Zeng.)*

Zhu Xiao, Xingxia Dai, Hongbo Jiang, Dong Wang, Liang Yang, and Fanzi Zeng are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China (e-mail: zhxiao@hnu.edu.cn; xingxdai@hnu.edu.cn; hongbojiang2004@gmail.com; wangd@hnu.edu.cn; liangy@hnu.edu.cn; zengfanzi@hnu.edu.cn).

Hongyang Chen is with the Institute of Industrial Science, University of Tokyo, Tokyo 153-8505, Japan (e-mail: dr.h.chen@ieee.org).

Digital Object Identifier 10.1109/JIOT.2019.2960631

I. Introduction

NOWADAYS, vehicles are not only transportation tools that fulfill people's travel demand but also participants involved in the promising vehicular applications, such as Internet of Vehicles (IoV) [1], heterogeneous vehicular networks [2], autonomous driving [3], etc., The emergence of vehicular applications relies on a large amount of data acquisition and processing, and this imposes considerable computing burden on vehicles and service providers [4], [5]. Specifically, if vehicular applications are to be completed effectively, it is often inevitable to seek help from others, which is referred to as task offloading [6]–[9].

In the vehicular scenarios, cloud computing [10] is not a good choice for task offloading, since it accompanies by large delay and low efficiency [11]. Alternatively, mobile-edge computing (MEC) [12] becomes a promising solution due to its proximity to the mobile vehicular terminals and hence effectively reduces the task's transmission delay. Besides, many entities can act as the MEC servers, such as small cell base stations [13] and roadside units (RSUs) [14], which commonly exist in the vehicular networks. A set of MEC servers are distributed in the urban environment covering a wide range area, in which each MEC provides services for its serving vehicle users.

In spite of the above-mentioned advantages of MEC, one challenge is the limited computing power of MEC servers [11]. This leads to a set of contradictions on the supply and demand of computing resources. In particular, according to the observation on daily life, we find that the vehicle users are always more sensitive to the delay of task processing especially when they drive through the urban hot zones [15]. For instance, people tend to get upset at the traffic jam moment that often takes place in the hot zones. If they request vehicular infotainment applications [16] and the task cannot be processed in time, they will be very upset. As such, the MEC servers have to alleviate excessive workload through refusing, postponing, or queuing the offloading requests of vehicular tasks [17], which will inevitably deteriorate vehicle users' quality-of-experience (QoE). A promising way to solve this is the cooperation among MECs, which is able to balance the spatial workload distribution, and in the meantime make use of computing resources more efficiently [13].

Vehicular task offloading via MEC cooperation is inherently feasible, as the formation of the urban hot zone follows the

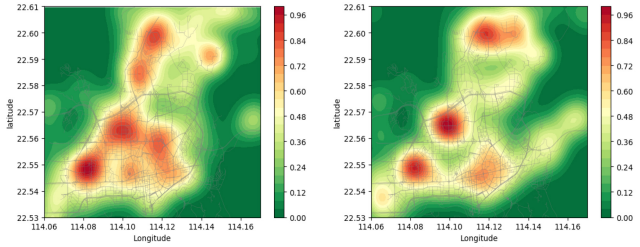


Fig. 1. Dynamic heat on September 12, 2018, in Shenzhen, China. (a) 9:00. (b) 15:00.

spatial–temporal aggregation effect [18]. This can be revealed by two observations. First, the MEC servers in hot zones may face unaffordable requests of computing resources. When people drive through the hot zone at a certain time period, they are likely to drive very slowly and even get stuck in the congested traffic, because the heat value maintains large. Higher heat indicates larger vehicle density and more requirements of vehicular application services due to the aggregation effect [18]. Fig. 1 illustrates the heat map embedded into the road networks via retrieving stop-and-wait points from large-scale private car trajectory data [18], in which x-axis and y-axis denote the range of the selected district in Shenzhen, China. The bar graph on the right in Fig. 1(a) and (b) describes the dynamic heat value. The deep red denotes large heat and hence represents the hot zone. Second, in the urban area, a hot zone is always surrounded by some *non*-hot zones. As seen in Fig. 1, the red areas (with high heat value), light red areas, and light green areas (with low heat value) are adjacent to each other. In other words, there will be some low-heat MEC servers that have redundant computing resources around the high ones, thereby providing a realistic foundation for cooperative vehicular task offloading among all MECs.

Nonetheless, we emphasize that the vehicular task offloading via MEC cooperation is not straightforward and we are facing several challenges.

1) *The Dynamic Changing of Heat in the Urban Hot Zones:* Fig. 1(a) and (b) presents the changing of the heat, which essentially demonstrates the time varying of the hot zones, namely, a hot zone at present may turn into a *non*-hot zone in the future. As a result, an MEC server that needs help from its neighboring MECs at the current moment may turn to be a helper at the next time slot in the task offloading. Therefore, it is crucial to obtain prior heat information to assist in determining whether an MEC can help its neighbors or needs help from others by cooperative task offloading.
2) *The MECs Role Determination Under Delay-Sensitive Constraint:* Given the heat information, to determine the roles of MECs, i.e., whether an MEC can help task offloading for its neighbors or needs help from others, the completion time of the tasks needs to be taken into ful consideration. The reasons are twofold. On one side, the vehicles move quickly and hence they are sensitive to the delay of vehicular applications; on the other side, when the vehicles have to drive slowly through the traffic jam and note that traffic congestion tends to occur in urban hot zones [15], the users exhibit low tolerance of the delay of task processing when they require vehicular services [19]. Therefore, how to minimize the time-delay during cooperative task offloading and in the meantime achieve efficient allocation of workloads among MECs with QoE guaranteed is an open issue in the vehicular MEC systems.
3) *The Grouping Problem in the MEC Cooperation:* The principle of the vehicular task offloading via MEC cooperation is that the MEC servers form nonoverlapping groups, each group consists of several MECs, in which the low-heat MEC helps its neighbors that are with high heat value. Provided that the roles of MECs have been identified, the choices of the MEC grouping are complicated. In addition, the grouping shows the exponential power 2 growth with the increase of the number of related MECs. In other words, how to form proper MEC groups is a considerable problem, in order to maximize the overall benefits and solve the MEC cooperation under a series of constraints.

In this article, we strive to realize vehicular task offloading through cooperation among MEC servers based on the analysis of hot zone. Specifically, our goal is to provide a collaborative way minimizing vehicular application latency (i.e., improving the users' QoE) while making the overall energy consumption of MEC servers as small as possible. The main contributions of this article are summarized as follows.

1) We capture the time-varying heat value of the hot zones via designing a heat prediction method. To specify, in our previous work, we have collected a large-scale private car trajectory data set in real-world urban scenario [20]. By extracting stop-and-wait points from the private car trajectories, we establish the association between the stop-and-wait data and the hot zone, and then design a prediction method based on the gated recurrent unit (GRU) networks to obtain the heat information of the hot zones at the next time slot. The heat prediction results can guide the role determination of MECs and ensure that cooperation among them being carried out successfully. To further identify the role of MEC, we take the time-delay constraint into consideration for the task offloading. Specifically, we design the workload allocation algorithm under the task delay constraint (detailed in Section IV-C).
2) In the MEC cooperation, MECs have less incentive to cooperate. Since each MEC is selfish, it hopes to maximize its own individual interests, which in turn greatly lowers the performance of others. Theoretically, we formulate the MEC grouping as a utility maximization problem via designing a noncooperative game-theoretic strategy selection based on regret-matching. Furthermore, we derive the correlated equilibrium and prove that the fast convergence can be achieved. Besides, we propose the algorithms based on the noncooperative game to obtain the strategy set and implement MEC cooperation, detailed in Section V-C, in the meantime minimize MEC energy consumption. The

proposed algorithms can effectively alleviate the conflicts between delay constraint and energy consumption.

3) Extensive simulation results validate the effectiveness and robustness of the proposed vehicular task offloading approach under various system parameters, such as traffic size, computation workload, time slots, and MEC servers number. We have demonstrated that the proposed method can significantly reduce the task complete delay, and in the meantime enhance the MEC energy efficiency with end users' QoE guaranteed.

The remainder of this article is organized as follows. Section II reviews the related works. Section III gives the system model. Section IV presents the heat-aware MECs role determination. In Section V, we describe the noncooperative game-theoretic method with correlated equilibrium. In Section VI, we present the evaluations, followed by the conclusions in Section VII.

II. Related Works

The proposal of mobile cloud computing has effectively alleviated the pressure of insufficient computing power of mobile terminals due to the increase in data volume. While cloud servers are far away from the end devices, hence not a suitable offloading object for delay-sensitive applications.

In recent years, MEC, which is overlapping and interchangeable with fog computing in many fields [21], has become a more feasible approach, since it is in proximity to the terminal equipment. There are many research works in terms of the fog radio access network (F-RAN) [22]–[26] and MEC task offloading [13], [17], [27]–[29]. Peng *et al.* [22], [25] and Xiang *et al.* [26] considered fog radio access networks in the 5G wireless communication system. They offered comprehensive studies on the key techniques for access slicing in F-RANs and corresponding solutions of F-RAN, the performance improvement and resource allocation optimization with the help of F-RANs. Related architectures were proposed and open issues were analyzed. The paper in [23] presented a mobile virtual reality delivery framework based on fog radio access networks, which aims at maximizing average tolerant delay under a serial of constrains by combining cache in advance with offloading. Zhao *et al.* [24] proposed a computation offloading scheme in fog radio access networks, and a joint optimization algorithm to minimize the total cost. The common offloading patterns include 1 to n [27] and n to n [17], [30]. Delay and energy consumption [31] are two main aspects considered during the offloading process. For example, in [28], the delay and energy consumption of task offloading are well traded off under two different scenes. In [29], offloading decision was made based on the prediction of execution time and energy consumption.

With the development of technologies, such as wireless communications, mobile sensing, and Internet of Things (IoT), the progress of vehicle networking technology has been accelerated. Vehicles can connect with vehicles (V2V), surrounding infrastructure (V2I), and others. In other words, vehicles can connect with everything (V2X) [32]. IEEE 802.11p in vehicular *ad hoc* networks (VANETs) and 5G networks are widely adopted in V2X networks [33]. On the one hand, vehicle connectivity supports a variety of emerging vehicular applications, such as autonomous driving and video-aided real-time navigation. On the other hand, massive data transmission poses a burden on radio access networks [34]. Du *et al.* [35] used vehicular terminals white space bands to supplement the bandwidth to solve the problem. Furthermore, vehicular communication systems are of high mobility and limited computing power [36], and the development of emerging vehicular applications leans on the powerful process equipment. Therefore, an MEC enabled vehicular networking is usually considered. In [37], vehicles make task offloading decisions to minimize the average response time for vehicle tasks. Zhang *et al.* [38] considered that in the case of a one-way street, vehicles with computation-intensive tasks can choose RSUs, which act as MEC servers in the situation, to undertake offloading workload according to the prediction of the current RSUs' computing resources, for the purpose of minimizing overhead. The vehicle and the chosen RSU communicate via V2V, or can be directly connected by V2I.

It is noteworthy that a single MEC server may not be able to meet computing needs when workload is too large. In order to make full use of the computing resources in the MEC servers, some works explore the cooperation relationship among MECs [13], [17]. The basic idea of cooperation among MEC servers is that some MEC servers may carry computations that exceed their computing capacity due to the uneven spatial distribution of computing workload, while some other MEC servers have redundant computing resources. MEC servers can help each other through further offloading so that they improve the overall task processing efficiency. Nevertheless, few works pay attentions on the specific matching method during the cooperation process, while it has great effects on the final offloading decision.

According to [13], there exists an upper limit of MECs energy consumption, and it would increase when cooperation is applied among MECs. Therefore, the final offloading decision should consider the constrain of energy consumption. As for the objective function, existing works give different optimal solutions, such as interior-point method [39], reinforcement learning [30], and game theory [40]. The game theory approach usually attributes to finding an equilibrium point. Chen *et al.* [41] adopted a game theory method to solve the problem of multiuser computation offloading and a Nash equilibrium is achieved. Liwang *et al.* [42] proposed a two-player Stackelberg-game for vehicular cloud computing in a 5G cloud-enabled scenario and a Stackelberg equilibrium was derived. Besides, there are works aiming to solve problems with correlated equilibrium [43]. For example, the correlated equilibrium was used to solve the allocation problem of spectrum resources in the LTE system [44].

In this article, when a nonhot MEC receives help requests from multiple hot MECs, this nonhot MEC selects to help one hot MEC, and its remaining computing resources to help other MECs would be reduced. Apparently, different choices

TABLE I
COMPARISON WITH EXISTING WORKS

Feature \ Method	[10]	[13]	[17]	[27]	[30]	[45]	[46]	this paper
Offloading pattern	n to n	n to n	n to n	1 to n	n to n	1 to n	n to n	n to n
Cooperation among MECs	no	yes	yes	no	yes	no	no	yes
Energy consumption of MECs	no	yes	no	no	yes	no	no	yes
Arbitrary divided tasks	yes	yes	yes	no	yes	yes	yes	no
Real world data	no	no	no	yes	no	no	no	yes
Temporal correlation	yes	yes	yes	yes	no	yes	yes	yes
Strategic behavior	yes	yes	no	yes	yes	no	no	yes

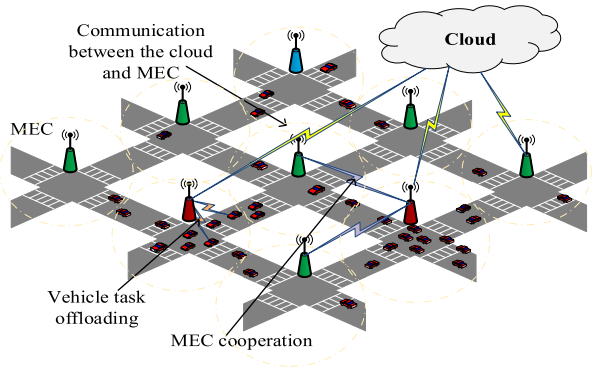


Fig. 2. System model.

will affect each other and will produce different MEC energy consumption. Therefore, it is required to come up with a final decision by integrating the overall information and relevant characteristics among players (i.e., MECs) and thereby minimizing task delay and energy consumption. Compared with Nash equilibrium, correlated equilibrium is more concerned with the relevant characteristics between participants [44], which is more in line with our target scenario. We can achieve the final MEC grouping relationship through the correlated equilibrium. Table I presents the comparison with existing works.

III. SYSTEM MODEL

In this section, we consider an edge computing ecosystem in the context of IoV. Fig. 2 presents the system model in the urban scenario, in which M MEC servers, indexed by $\mathcal{M} = \{1, \ldots, M\}$, are distributed in the urban scenario. The moving vehicles, denoted by $\mathcal{N} = \{1, \ldots, N\}$, can communicate with different MEC servers in different time slots, and each MEC can only connect with vehicles within its coverage. As illustrated in Fig. 2, the red MEC servers take care of the hot zones, where a large number of vehicles drive through. Besides, the red MEC servers are with some adjacent green ones that serve a few vehicle users. The cloud server is used to provide a centralized way to make prediction on the amount of vehicle tasks based on the historical real-world vehicular stop-and-wait points data. This indicates that the green MECs serving nonhot zones can help task offloading for the red ones. For a task v of vehicle n, $\mathcal{V}_n^v \triangleq (B_n^v, C_n^v, D_n^v)$, $v \in \{1, \ldots, V\}$, where B_n^v, C_n^v, and D_n^v is the data size, computation workload, and deadline requirement of the task v for the vehicle n, respectively. Task v can be offloaded to the MEC servers in order to reduce the completion delay. The main notations used in this article are summarized in Table II.

A. Task Model

Different from the models which divide a task into different parts arbitrarily [17], we consider the sequential dependency between components which makes up of the task. Components are described by a directed weight graph $\mathcal{G} = (u, \varepsilon)$, where u represents the required computing resources when the component is executed, ε denotes the amount of exchanged data between adjacent components, i.e., the serial dependency. The values of (u, ε) are different for different components. W_v is used to denote the number of components that make up of the task v. According to Jia *et al.* [47], continuous components offloading for a task can achieve near-optimal offloading decisions and minimize the delay. Thus, vehicular offloading decisions are based on finding the start component and the end one, i.e., J_0 and J. A task is offloaded to the MEC servers from component J_0, and back to be processed locally from component J, offloaded components are executed in order. For the overloaded MEC servers, parts of tasks can be offloaded to their adjacent MEC servers based on the idea of MEC cooperation, thus $J = \{J_1, \ldots, J_m\}$, $J_1, \ldots, J_m \in \{J_0, \ldots, W_v - 1\}$, $m \in \mathcal{M}$. $J_1 = J_m$ means a task is only offloaded to an MEC, which covers the vehicles in its transmission range. The u constitutes computing workload of task v for vehicle n, and influences computing delay. The ε values of components J_0 and J for a task v determine the data size of the task v, which influences the transmission delay. The total MECs energy consumption is based on each MEC workload obtained by minimizing tasks complete time. As a result, (u, ε) affect the performance of MEC energy consumption. Offloading decision is indicated by $\alpha_v^t \in \{0, 1\}$, which means task v is executed locally in time slot t when $\alpha_v^t = 0$. Otherwise, some components are offloaded. β_v^t is used to indicate whether cooperative offloading (CO) happens for the task v in time slot t.

B. Communication Model

Communications occur on the wireless link between vehicles and MEC servers. Vehicles first offload some components to the MEC servers for each task in order to minimize the processing time, then the computational results should be returned. For the first process, we obtain the data rate of the computation offloading between the vehicle n and the MEC m according to the Shannon capacity, $r_{mn}^t = B\log_2(1 + ([H_n^m P_n^m]/\sigma^2))$, where r_{mn}^t and B denote transmission rate and channel bandwidth, respectively. H_n^m and P_n^m are channel gain and transmission power between the vehicle n and the MEC

TABLE II
MAIN NOTATIONS

Notation	Description
B_n^v	data size of task v of vehicle n
C_n^v	computation workload of task v of vehicle n
D_n^v	deadline requirement of task v of vehicle n
W_v	components number that make up of the task v
α_v^t	an indicator denoting whether the offloading occurs for task v at time t
β_v^t	an indicator denoting whether cooperative offloading occurs for task v at time t
(J_0,J)	offloading components locations for task offloading
r_{mn}^t	transmission rate between vehicle n and MEC m in time slot t
T_{trans}^{mn}	transmission delay
$\mathscr{L}_i^j$	relative positions of the vehicle and returned computing results
T_{comp}^{vt}	computing delay
f^{local}	computing ability of vehicles
f^m	computing ability of MEC servers
ε_{comp}^{mt}	computing energy consumption of MEC m
W_m^t	vehicles number under MEC m coverage in time slot t
ε_{launch}^t	launch energy consumption
δ^t	the number of MECs act as a helper in time slot t
T_{queue}^t	queuing delay
ϕ^t	total workload offloaded by cooperation among MEC servers
$\varepsilon_{schedule}^t$	scheduling energy consumption
ϖ^t	hot MECs number in time slot t
M^t	MECs number participating in cooperative offloading in time slot t
$\mathscr{G}_m$	normalized computing capacity of MEC m
$\mathscr{H}_m^t$	normalized workload of MEC m at time slot t
$\widetilde{M^t}$	MECs number participating in cooperative offloading in time slot t
$\widetilde{A_m^t}$	the strategy set of MEC m in time slot t
$\widetilde{\delta_m^t}$	MECs number providing help for MEC m in time slot t
$\widetilde{U_m^t}$	the utility set of MEC m in time slot t
$D_m^T(i,j)$	utility difference for adopting different behavior
$R_m^T(i,j)$	degree of regret for adopting different behavior

m, and σ^2 represents noise power. We assume that the transmissions are operated on orthogonal channels. Consequently, the delay of transmission is given by

$$T_{\text{trans}}^{mn} = \frac{B_n^v}{r_{mn}^t}. \tag{1}$$

For the second process, vehicles move quickly and drive across the coverage area of different MECs. $\mathscr{L}_i^j$, $i, j \in \mathscr{M}$, represents the vehicle that is currently within the coverage of MEC i, while the last offloaded component J_m is processed in MEC j. Expression $i \neq j$ means the vehicle and results should be returned to different MEC servers, and notes that they cannot communicate directly. Under these circumstances, the cloud server performs as a relay and solve this information exchange problem. Furthermore, we do not consider delay and energy consumption of result transfer, since the data size of the result is much smaller than the one before processing, and the downlink rate from an MEC or the cloud server to a vehicle is higher than the uplink rate from a vehicle to an MEC.

C. Computation Model

We discuss the computing delay and energy consumption of MECs.

1) *Computation Delay:* In the following, we present the definitions of computation delay and energy consumption

$$T_{\text{comp}}^{vt} = \frac{C_n^v}{f^*} \tag{2}$$

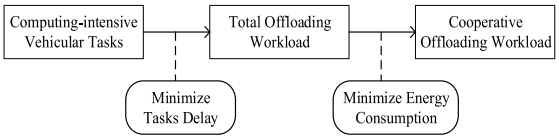


Fig. 3. Procedure of task offloading.

where

$$f^* = \begin{cases} f^{\text{local}} \\ f^m, \quad m \in \mathscr{M} \end{cases}$$

represents the computing ability of the vehicles and MEC servers, respectively. The computation delay determines the task computing time, and exerts a direct influence on the vehicular task complete time. This is related to each MEC workload obtained by minimizing tasks complete time. Accordingly, f^{local} and f^m affect the performance of MECs energy consumption.

2) *Energy Consumption of MEC m:*

$$\varepsilon_{\text{comp}}^{mt} = \sum_{n \in W_m^t} \sum_{v \in V} k\left(f^m\right)^2 C_n^v \tag{3}$$

where k is the energy coefficient and relies on the chip architecture, $n \in W_m^t$, and W_m^t represents vehicles number under MEC m coverage in time slot t. Fig. 3 presents the procedure of task offloading. Through the minimization of task delay and MECs energy consumption, the final task offloading decision is achieved.

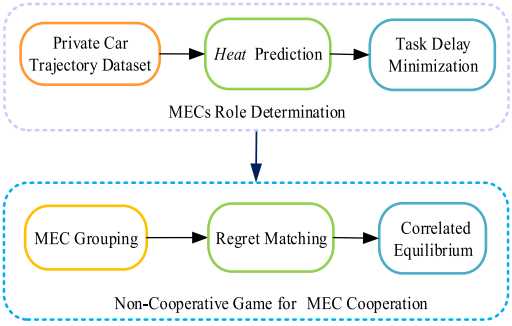


Fig. 4. Block diagram of the proposed method.

IV. Heat-Aware MECs Role Determination in Cooperative Task Offloading

In this section, we present the heat-aware MECs role determination. The noncooperative game for MEC cooperation will be discussed in the next section. Fig. 4 illustrates the framework of the proposed method.

A. Observations

Observation 1: The high heat value means high vehicle density, and the number of potential vehicles that request task offloading is large. What is more, traffic congestion often occurs in hot zones with high heat value [15], and the vehicle users tend to request much more vehicular application services. In turn, this, on the one hand, further increases the computing burden of MECs, on the other hand, significantly reduces QoE of the vehicle users. In such cases, higher task latency requirements should be met, since vehicle users are generally anxious because of traffic congestion, and they become more sensitive to the processing delay of tasks.

Observation 2: Recall Fig. 1, we observe that there are some MECs with low heat value locating close to the overloaded MECs. This indicates that, first, there will be unused or redundant computing resources for the MECs in nonhot zones; second, these MECs could help task offloading for the overloaded MECs under the cooperation mechanism.

In the following, we obtain the expression of task delay and MEC energy consumption during the process of cooperative task offloading. MECs in nonhot zones will process some additional workloads due to the cooperative task offloading. Although MECs connection is wired, we do not consider the transmission energy consumption among them [13]. In addition, there are energy consumption produced for taking a longer time than the original to occupy the communication channel between them, which is called launch energy consumption [48]

$$\varepsilon^t_{\text{launch}} = \lambda \delta^t \tag{4}$$

which is proportional to the number of MECs providing assistance. λ is the coefficient, and δ^t represents the number of MEC act as a helper in time slot t. When an MEC provides offloading assistance for several MEC servers in hot zones, there will be queue delays because of network congestion caused by the limited bandwidth of MEC and the uncertainty of the workload arrival of vehicles. $M/M/1$ queuing model is a widely used method to characterize this process and the corresponding queuing delay is given as

$$T^t_{\text{queue}} = \frac{\gamma}{1 - \gamma \phi^t} \tag{5}$$

where γ is the expected delay for sending and receiving data without considering the network congestion, and ϕ^t is the total normalized workload offloaded by cooperation among the MEC servers. Note that MECs in the nonhot zone have to help several adjacent hot MECs (1 to n) and an MEC in the hot zone may need assistance from its adjacent nonhot MECs (n to 1). Under these circumstances, scheduling for workload is necessary and the energy consumption is produced [49], [50]

$$\varepsilon^t_{\text{schedule}} = \rho \varpi^t \tag{6}$$

which is related with ϖ^t, the number of the hot MEC servers in time slot t, and ρ is the coefficient. We have $M^t = \delta^t + \varpi^t$, which represents MEC number participating in the cooperative task offloading.

B. Heat Prediction Based on Real-World Trajectory Data

Along with the development of urbanization and motorization, in recent years, we have witnessed the continuous growth of the ownership of automobiles especially the private cars, which leads to a direct factor contributed to the change of travel mode for the urban residents [51], i.e., as a convenient way to commute and travel, more and more people choose to buy a car to fulfill their daily travel needs.

In particular, when people drive and arrive at certain hot zones, such as places of work and shopping areas, these places are likely to be visited by a number of vehicles at specified moments. Such travel behavior can be observed via studying the stop-and-wait points retrieving from private car trajectory data [18], which essentially reflects the heat information of urban hot zones. In our previous work, we have developed the vehicle state estimation method [20] and applied it to collect the large-scale private car trajectory data set [52]. This motivates us to come up with a prediction method so as to estimate the future heat for the specified area based on the analysis of the historical private car stop-and-wait data. Apparently, the predicted heat results will guide the action identifying the role of MECs, i.e., whether an MEC is a helper or needs help in the cooperative task offloading.

Accordingly, we design a prediction method based on the GRU networks to obtain the heat information of hot zones at the next time slot, with the aims at capturing the time-varying heat value of the hot zones and guiding the role determination of MECs. Unlike the long short term memory (LSTM), GRU only has two gates: 1) reset gate and 2) update gate [53]. The hidden layer state of GRU is the linear interpolation between the previous state and the candidate state at t

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot H_t. \tag{7}$$

Update gate is expressed by z_t, which can be regarded as a combination of forget gate and input gate in the LSTM loop body

$$z_t = \zeta(W_{xz} x_t + W_{hz} h_{t-1} + b_z). \tag{8}$$

The update gate determines how many previous state information can enter the current time, and the closer z_t is to 1, the more the current state uses the previous time information

$$H_t = \tanh\left(W_{x_t} + U(r_t \odot h_{t-1}) + b_h\right) \tag{9}$$

where H_t is the candidate state, r_t is the reset gate, and $\odot$ represents element-wise product of two vectors

$$r_t = \zeta(W_{xr}x_t + W_{hr}h_{t-1} + b_r). \tag{10}$$

The closer r_t is to 0, the smaller the proportion of the output state at the previous moment.

Through the GRU prediction method based on the real-world private car stop-and-wait data set, we can characterize the dynamic process of the varying heat, and thereby reckoning the vehicle density and offloading vehicular tasks.

C. Task Delay Minimization in Workload Allocation

In the process of vehicular task offloading, the total delay for the completing task v includes: 1) transmission delay between the vehicles and MEC servers; 2) queue delay caused by network congestion; and 3) computing delay for the task processing at different platform. We formulate the task delay minimization problem for achieving efficient workload allocation

$$\begin{aligned}
\text{minimize}\quad & T^{mt}_{\text{complete}} \\
&= \sum_{n\in W^t_m}\sum_{v\in V}\Bigg(T^{vt}_{\text{comp}} + \alpha^t_v\Bigg(T^{mn}_{\text{trans}} + \beta^t_v T^t_{\text{queue}} \\
&\qquad + \left(T^t_{\text{comp}}\right)' + \beta^t_v\left(T^t_{\text{comp}}\right)''\Bigg)\Bigg) \\
&= \sum_{n\in W^t_m}\sum_{v\in V}\Bigg(T^t_{\text{comp}} + \alpha^t_v\Bigg(T^{mn}_{\text{trans}} + \left(T^t_{\text{comp}}\right)' \\
&\qquad + \beta^t_v\left(T^t_{\text{queue}} + \left(T^t_{\text{comp}}\right)''\right)\Bigg)\Bigg) \\
&= \sum_{n\in W^t_m}\sum_{v\in V}\Bigg(\frac{C^v_n}{f^{\text{local}}} + \alpha^t_v\Bigg(\beta^t_v\Bigg(\frac{\gamma}{1-\gamma\phi^t} + \sum_{k\in\mathscr{L}^t_m, k\neq m}\frac{C^v_n}{f^k}\Bigg) \\
&\qquad + \frac{B^v_n}{r^m_n} + \frac{C^v_n}{f^m}\Bigg)\Bigg) \qquad (11)
\end{aligned}$$

$$\text{s.t.}\quad T^{mnvt}_{\text{complete}} \le D^v_n \tag{12}$$

$$\alpha^t_v \in \{0,1\}, v \in \mathscr{V} \tag{13}$$

$$\beta^t_v \in \{0,1\}, v \in \mathscr{V} \tag{14}$$

$$\varepsilon^t_{\text{complete}} \le E. \tag{15}$$

$(T^t_{\text{comp}})'$ and $(T^t_{\text{comp}})''$ represent the offloading computing delay and the further computing offloading delay for MECs cooperation, respectively. Constraint (12) denotes the upper bound of the task complete delay. Equations (13) and (14) represent whether offloading and CO exist, respectively. Constraint (15) gives the limitation to the total energy consumption.

$\mathscr{G}_m$ and $\mathscr{H}^t_m$, $m \in \mathscr{M}$, are used to represent normalized computing capacity and workload of MEC m at time slot t, respectively. According to $\mathscr{G}_m$ and $\mathscr{H}^t_m$, we classify MEC servers into the following categories.

1) *Hot MEC ($\mathscr{A}$):* An MEC is a hot MEC if it offloads a positive portion ε_1 of its computing workload to the adjacent MEC servers and executes the rest of workloads. Moreover, it does not receive any workload from other MEC ($0 < \varepsilon_1 \le 1$, $\mathscr{G}_m < \mathscr{H}^t_m$).
2) *Neutral MEC ($\mathscr{R}$):* An MEC is a neutral MEC if it neither seek help from other MEC servers or help to them. $\mathscr{P}^t_i$ is the adjustable lower value ($\mathscr{P}^t_m < \mathscr{H}^t_m \le \mathscr{G}_m$).
3) *Nonhot MEC ($\mathscr{N}$):* An MEC is a nonhot MEC if it receives workload from other MEC servers and does not offload workload to others ($\mathscr{H}^t_m < \mathscr{G}_m$).

Algorithm 1 presents the processing of heat-aware MECs role determination. Algorithm 1 contains two parts, the first part obtains offloading workload based on minimizing vehicular tasks complete delay, its complexity is $O(TV(W^v-2)^2)$, where T is the number of time slots, V represents vehicular tasks number, and W^v is used to denote the number of components that make up of the task v. The second part of Algorithm 1 determines the heat-aware MECs role based on obtained offloading workload, its complexity is $O(TM)$, where M denotes MECs number. Thus, the complexity of Algorithm 1 is $O(TV(W^v-2)^2 + TM)$. For the reason that $V(W^v-2)^2 > M$, its complexity is expressed as $O(TV(W^v-2)^2)$.

V. Noncooperative Game-Theoretic Method With Correlated Equilibrium

After MEC role being determined, we will figure out how to form the proper MEC groups, i.e., to resolve the MEC grouping problem in the processing of MEC cooperation. First, the effective MEC grouping should meet the following conditions: 1) the proximity principle, namely, the hot MEC servers overload can only be assisted by the adjacent nonhot MEC servers and 2) the nonhot MEC servers cannot exceed their own computing tolerance when they provide task offloading to the hot ones. In the vehicular task offloading with the game-theoretic method, MECs in hot zones are regarded as players in the game, they make grouping decisions individually for minimizing their own energy consumption, regardless of whether or not hurt the overall interest ultimately. In this regard, the MEC is selfish [43], [44].

In this section, we propose a noncooperative game-theoretic method with correlated equilibrium to solve the MEC grouping problem. To specify, within the proposed method, the MECs make the grouping decisions and obtain the optimal strategy based on the regret matching, with the aims at improving the energy efficiency and in the meantime enhancing the users' QoS.

A. Noncooperative Game for MEC Grouping

We formulate the MEC grouping problem as a noncooperative game

$$\Gamma = \left\{\widetilde{M}^t, \left\{\widetilde{A}^t_m\right\}, \left\{\widetilde{U}^t_m\right\}\right\} \tag{16}$$

Algorithm 1 Heat-Aware MEC Role Determination

Input: $\mathscr{G} = (u,\varepsilon)$, $\mathscr{G}_m$.
Initial Settings: $J_0 = J_m = 2$, $T_{mim} = T^{mt}_{complete}(2,2)$.
Output: MEC role $a[M]$.
for $t = 1, 2, ..., T$ **do**
  for $v = 1, 2, ..., V$ **do**
    for $i = 2, ..., W^v - 1$ **do**
      for $j = i, i+1, ..., W^v - 1$ **do**
        According (12) to (15), making offloading decisions for minimizing task delay.
        if $T^{mt}_{complete}(i,j) < T_{mim}$ **then**
          $J_0 = i$.
          $J_m = j$.
          $T_{mim} = T^{mt}_{complete}(i,j)$.
        end if
      end for
    end for
  end for
  According (7) to (10), adopting GRU prediction to obtain heat value of different zones.
  Obtaining offloading workload $\mathscr{H}^t_m$.
  for $i = 1, 2, ..., M$ **do**
    if $0 < \varepsilon_i \le 1$ and $\mathscr{G}_i < \mathscr{H}^t_i$ **then**
      $a[i] = 1$.
    end if
    if $\mathscr{P}^t_i < \mathscr{H}^t_i < \mathscr{G}_i$ **then**
      $a[i] = 0$.
    end if
    if $\mathscr{H}^t_i < \mathscr{G}_i$ **then**
      $a[i] = -1$.
    end if
  end for
end for

where $\widetilde{M}^t$ is the finite set of the MEC servers involved in the MEC grouping at time slot t. $\widetilde{A^t_m} = \{A^t_{m1}, \ldots, A^t_{mk}\}$, denotes the strategy set of MEC m in time slot t, where $k = \delta^t$, $m \le \varpi_t$ and $m \subseteq \widetilde{M}^t$. $A^t_{mk} = 1$ means the nonhot MEC k provides help to the hot MEC m, otherwise $A^t_{mk} = 0$. $\widetilde{U^t_m}$ is the utility set of MEC m in time slot t, which is used to find the optimal strategy from the candidate strategies in the noncooperative game.

The main purpose of grouping among the MEC servers is to find the proper MEC groups for cooperative task offloading and in the meantime minimize the energy consumption. We define the reduced energy consumption as the reward of the utility function

$$U^t = \mu - \varepsilon^t_{\text{complete}} \tag{17}$$

where μ is a constant and to ensure U^t is positive. The total energy consumption for completing task v includes: 1) energy consumption for the nonhot MEC who acts as a helper; 2) energy consumption of scheduling when a not-hot MEC needs to help several adjacent hot MEC servers; and 3) computing energy consumption for task processing to different MEC servers. We formulate the following energy minimization problem:

$$\begin{aligned}
\text{minimize}\quad & \varepsilon^t_{\text{complete}} \\
&= \sum_{m\in M} \alpha^t_v \Big(\varepsilon^t_{\text{launch}} + \varepsilon^t_{\text{schedule}} \\
&\qquad + \Big(\varepsilon^{mt}_{\text{comp}} + \beta^t_v \big(\varepsilon^{mt}_{\text{comp}}\big)' \Big) \Big) \\
&= \sum_{m\in M} \sum_{n\in W^t_m} \sum_{v\in V} \alpha^t_v \Bigg(\lambda\delta^t + k\big(f^m\big)^2 C^v_n + \rho\varpi^t \\
&\qquad + \sum_{k\in \mathscr{L}^t_m, k\ne m} \beta^t_v k\big(f^m\big)^2 C^v_n \Bigg) \qquad (18) \\
\text{s.t.}\quad & \sum_{m=1}^{M^t} \mathscr{H}^t_m \le \sum_{m=1}^{M^t} \mathscr{G}^t_m \qquad (19) \\
& \phi^t \le \sum_{m=1}^{\delta^t} \big(\mathscr{G}^t_m - \mathscr{H}^t_m\big) \qquad (20) \\
& \alpha^t_v \in \{0,1\}, v \in \mathscr{V} \qquad (21) \\
& \beta^t_v \in \{0,1\}, v \in \mathscr{V} \qquad (22) \\
& \varepsilon^t_{\text{complete}} \le E. \qquad (23)
\end{aligned}$$

$(\varepsilon^{mt}_{\text{comp}})'$ denotes the computing energy consumption caused by cooperation. Constraints (19) and (20) show the restrictions on CO workload. Equations (21)–(23) represent the same constraints in the optimization problem (13)–(15). We obtain strategy set based on (19)–(23), then choose the optimal strategy, namely, the optimal MEC grouping, according to the defined utility function.

B. Correlated Equilibrium

A related equilibrium of [43] is a probability distribution on the strategy combination

$$\sum_{m\in\widetilde{M}^t} \widetilde{U^t_m}\Big(A^t_{mi}, \widetilde{A^t_{-m}}\Big) - \widetilde{U^t_m}\Big(A^t_{mj}, \widetilde{A^t_{-m}}\Big) \le 0 \tag{24}$$

where $\widetilde{A^t_{-m}}$ is the strategy combination without MEC m, and $\widetilde{U^t_m}(A^t_{mi}, \widetilde{A^t_{-m}})$ represents the utility of MEC m whose strategy set includes the strategy A^t_{mi}. We assume that strategy A^t_{mi} is applied at time slot $T-1$, and we intent to replace it by A^t_{mj} at next time slot. According to regret-matching proposed in [43], we obtain the utility difference, which is given by

$$D^T_m(i,j) = \frac{1}{T}\sum_{t=1}^{T}\Big[\widetilde{U^t_m}\Big(A^t_{mi}, \widetilde{A^t_{-m}}\Big) - \widetilde{U^t_m}\Big(A^t_{mj}, \widetilde{A^t_{-m}}\Big)\Big]. \tag{25}$$

Furthermore, we define $R^T_m(i,j)$ as the degree of regret for A^t_{mi} rather than A^t_{mj}

$$R^T_m(i,j) = \max\big\{D^T_m(i,j), 0\big\}. \tag{26}$$

Based on (16), we derive the following probability distribution when strategy A^t_{mi} and A^t_{mj} are selected

$$P^{T+1}_m(i) = \frac{1}{\eta} R^T_m(i,j) \tag{27}$$

$$P^{T+1}_m(j) = 1 - \sum P^{T+1}_m(i). \tag{28}$$

Proof: We define the empirical distribution of all player played strategies till T is $\varsigma \in \varrho(\widetilde{A^T})$, for each $a^T \in \widetilde{A^T}$, we use $\rho(a^T, \widetilde{A^T})$ denote the occurrences of single strategy a^T during the period of T

$$\varsigma^T(a^T) = \frac{\rho\left(a^T, \widetilde{A^T}\right)}{T} \tag{29}$$

which shows the frequency of strategy a^T is adopted during the period. That T approaches to ∞ means ς^T converges to correlated equilibrium based on analysis in [43]. ■

C. Algorithms for MEC Cooperation

In the MEC cooperation, there are many choices to form the nonoverlapping MEC groups. To specify, each MEC in nonhot zones can choose to help its adjacent ones in hot zones, and each MEC in hot zones can choose which MEC in nonhot zones to help it. In a word, the strategy set is large and the individual strategy interacts with each other. Taking the 7×7 MEC distribution as an example, the strategy space at 8:00 and 12:00 has reached million options. According to (19)–(15), the original strategy space is constrained, especially on location and offloading workload. First, the position of grouped MEC servers must be adjacent. Then we compare the actual offloading workload for each hot MEC server and the maximum computing ability of each nonhot MEC, and obtain the amount of actual offloading workload that can be undertaken by the nonhot MEC servers in the group. The offloading workload is guaranteed no more than its computing ability. The details of obtaining the strategy set are presented in Algorithm 2. The computational complexity of Algorithm 2 is $O(TM)$.

Based on (25)–(28), the regret-matching is applied during the MEC grouping process, aiming at finding out the optimal choice in the strategy set obtained by Algorithm 2. The main idea of regret matching is to compare the utility function of current strategy and other strategies, i.e., to obtain the regret factor $R_T^m(i, j)$, which denotes the regret degree implying not to adopt other strategies currently. After that, the strategy with a positive regret factor, i.e., larger utility function, becomes the candidate strategy. The comparison repeats in the candidate strategy until the optimal strategy that meets the requirements is selected at the next search. Algorithm 3 presents the regret matching for the selection of the MEC grouping strategy. The computational complexity of Algorithm 3 is $O(T\varpi^t\mathscr{A}_m^t)$, where ϖ^t represents hot MECs number in time slot t, $\mathscr{A}_m^t$ is the strategy space obtained by Algorithm 2.

Algorithm 2 Obtaining Strategy Set

Input: MEC role $a[M]$, location information.
Initial Settings: $b[M^t]$, $c[\varpi^t]$, $d[\delta^t]$, $e[\delta^t] = \{0\}$.
Output: Strategy set $\mathscr{A}_m^t$
for $t = 1, 2, ..., T$ **do**
  for $i = 1, 2, ..., M$ **do**
    if $a[i] \neq 0$ **then**
      $b[r1] = \mathscr{G}_m - \mathscr{H}_m^t$, $r1 = 1, 2, ..., M^t$
    end if
  end for
  for $j = 1, 2, ..., M^t$ **do**
    if $b[j] < 0$ **then**
      $c[r2], r2 = 1, 2, ..., \varpi^t \leftarrow$ actual offloading workload for each hot MEC server.
    end if
    if $b[j] > 0$ **then**
      $d[r3], r3 = 1, 2, ..., \delta^t \leftarrow$ maximum CO computing ability of each nonhot MEC.
    end if
  end for
  for $k = 1, 2, ..., \delta^t$ **do**
    combining location information.
    $e[k] \leftarrow$ actual offloading workload undertaken by each not-hot MEC servers.
    if $e[k] \leq d[k]$ **then**
      strategy ++.
    end if
  end for
  Update strategy set.
end for

Algorithm 3 Regret Matching for Strategy Selection

Input: Strategy set $\mathscr{A}_m^t$ according to Algorithm 2.
Output: The convergent value.
for $t = 1, 2, ..., T$ **do**
  for $i = 1, 2, ..., \varpi^t$ **do**
    Obtain different utility according to (17).
    Calculate the degree of regret for A_{mi}^t is adopted rather than A_{mj}^t based on (26).
    for $j = 1, 2, ..., \mathscr{A}_m^t$ **do**
      Update action by (27) and (28).
    end for
    Update strategy set.
    Update degree of regret (26).
    Iteration, feedback to 1, until the convergence value is reached.
  end for
end for

VI. Performance Evaluation

In this section, we consider the selected area in Luohu District of Shenzhen, China, as mentioned in Fig. 1, and assume that the MEC servers are scattered over a 7×7 mesh region and the cloud server is located in the center of the area. In the experiments, we use the laptop with Intel Core i5-8400 CPU @ 2.80-GHz, 8-GB memory, Win10 OS.

The main parameters in simulations are listed in Table III.

Five methods are used as benchmarks in the performance evaluation.

1) *No Offloading (NO)*: Vehicles will not ask for help from the MEC servers and tasks are executed locally [17].
2) *No CO (NCO):* Offloading workload is executed by an MEC and there is no cooperation among MEC servers [54]. Therefore, even an MEC whose computing

TABLE III
PARAMETERS SETTING

Parameter	Value
Number of MEC servers	49
An MEC radius	150m
The cloud server radius	2500m
Transmit power for the vehicles P_n^m	10dBm
Transmit power for MEC servers	20dBm
Transmit power for the cloud server	40dBm
Noise power σ^2	-115dBm
Channel bandwidth B	5 MHz
Local computing capability f^{local}	1GHz
MEC computing capability f^m	10GHz
Transmission delay γ	200ms

resource is deficient, its adjacent MECs having excessive computing capacity would not provide an assistant for task offloading [54].

3) *CO:* CO exists, while the MECs role cannot be told in advance without prediction methods [11].
4) *Total Offloading (TO):* The components that make up the task are processed at the MEC server-side except few components which can only disposed locally, and severs' computing sources are assumed sufficient [30].
5) *Random Task Offloading (RTO):* The hot MEC randomly asks an adjacent not-hot MEC for help regardless of its workloads are effectively released.

A. Heat Prediction

In our previous work, we have collected large-scale private car trajectory data in real-world urban environments. In order to characterize the heat of the selected urban area, we calculate the density of the stop-and-wait points from private car trajectories collected in Luohu District of Shenzhen, China. The open access of the data set is provided in [20]. We adopt the GRU-based prediction method to further capture the dynamic time-varying heat information, which helps to implement the role determination of MECs in the next time slot.

We select the stop-and-wait data collected in Luohu District, Shenzhen, China from August 20, 2018, to September 9, 2018, namely, use three-weeks trajectory data as the training set. Fig. 5(a) and (b) present the heat prediction results at 15:00 on September 12, 2018, which are obtained by using the support vector regression (SVR) method and our proposed GRU-based prediction method, respectively. Recall Fig. 1(b) in Section I, which gives the heat distribution. Comparing the results in Fig. 5 with Fig. 1(b), it can be seen that our proposed GRU-based prediction method obtain more accurate heat information than that by using SVR. Moreover, the results of the GRU prediction are roughly consistent with the groundtruth shown in Fig. 1(b), especially the formation of hot zones, which thus provide valuable information for the role determination for the heat-aware MEC cooperation.

B. Task Delay

Figs. 6 and 7 present the task delay with computation workload and traffic size, respectively. As shown in Fig. 6, task delay increases with the growth in computing workload. When the amount of offloading workloads is small, the task delays

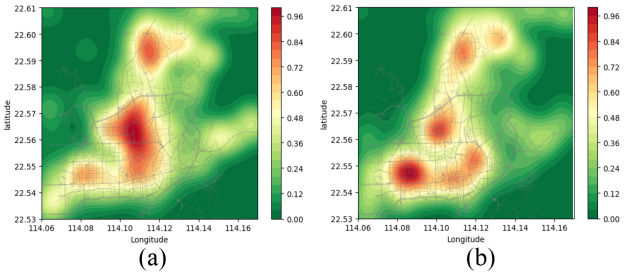


Fig. 5. Predicted *heat* at 15:00 on September 13, 2018, in Shenzhen, China. (a) SVR. (b) GRU-based prediction.

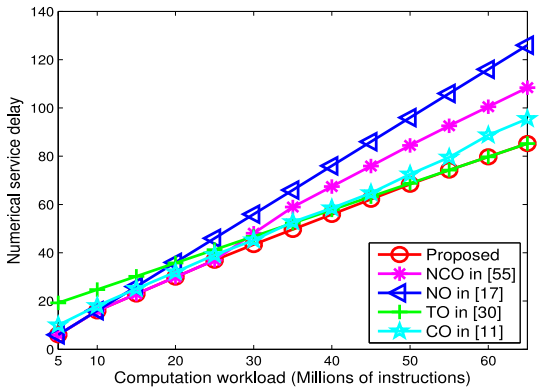


Fig. 6. Task delay with computation workload.

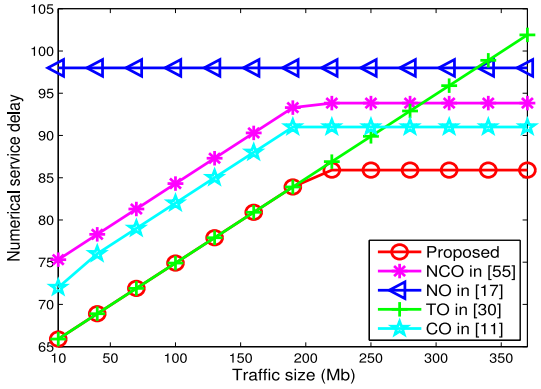


Fig. 7. Task delay with traffic size.

of NCO and the proposed are small while TO's delay is large. This is because it will be an optimal choice that tasks are executed locally when computing workload is small for saving time cost on transmission and queuing. Furthermore, when tasks are of a great deal of computing workload, the proposed method can still achieve low task delay as TO does. However, NO has the largest delay. CO is unable to perform as well as the proposed, since without prediction of the prior information on heat, using the CO method would degrade the performance of task cooperation. Therefore, when the requested computing resource is more than the local computing power, and even exceeds the computing capacity of an MEC, the cooperation among MECs provides an effective method for task offloading.

Fig. 7 presents that the task delay increases with the rising of traffic data. We find out that delays of TO and the proposed method are rather low as data size is small, while TO's delay

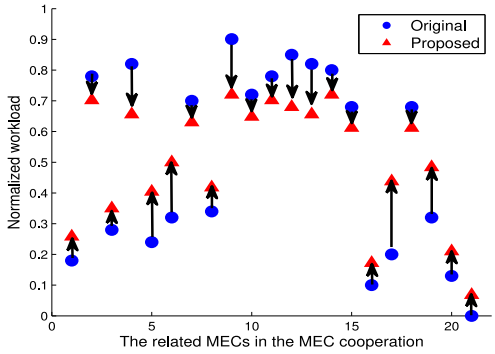


Fig. 8. Effect of cooperative task offloading.

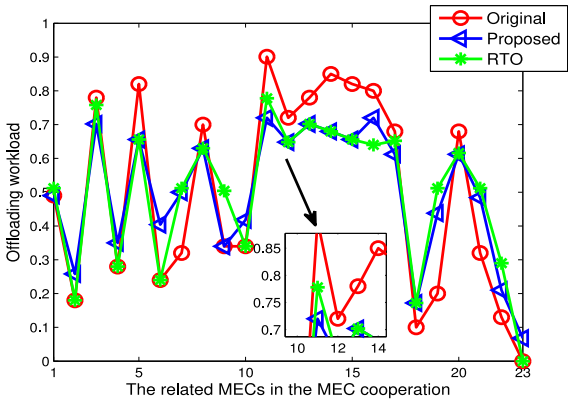


Fig. 9. Computing workload with various methods.

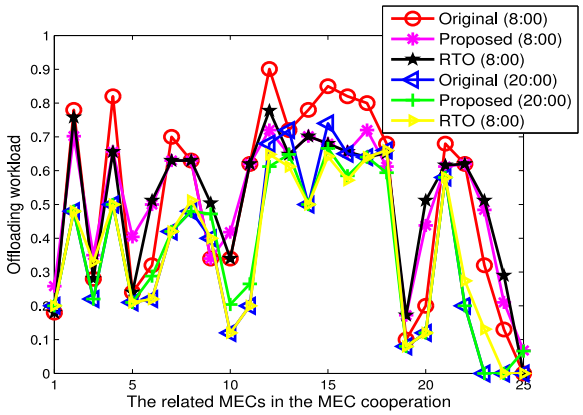


Fig. 10. Computing workload on different time slots.

turns to be the highest one when traffic size becomes large. The reason behind this is that the bigger of data size, the more transmission delay and even exceeds the computation time saved by offloading. Besides, we note that the requested computing workload is set so big that it cannot be undertaken by the vehicle itself and MEC, thus the complete time of NO and TO is high. The CO method cannot make accurate judgments on the MEC roles at the next moment and thus fails to provide effective guidance for the cooperation stage. As a result, the performance of CO is not as good as the proposed method. According to the above analysis, we conclude the proposed method outperforms the comparative methods in terms of reducing task delay.

C. Evaluation of Cooperative Task Offloading

Fig. 8 shows an example that presents the normalized workload changes for the MEC servers whether or not applying cooperative task offloading at 8:00. The x-axis "the related MECs in the MEC cooperation" represents MECs participating in the cooperation and we sort them from number 1. Note that the related MEC servers number is less than M, namely, the total number of MEC servers. Some MECs would not participate in the process of collaborative grouping, which means that the workload they have to undertake will not change, and we do not mark them here. The y-axis "offloading workload" denotes an MECs workload, which reflects the delay required by the MEC server to process vehicular tasks. We normalized the workload in order to correspond to the prediction map in Fig. 5. It can be seen that there are several MEC servers whose workloads are very high such as the ninth MEC server, its normalized value is even higher than 0.9. It means that this MEC needs to carry more computing loads than its actual computing capacity. Thus, congestion, delay, and even loss for these offloaded data will take place, which inevitably deteriorates the quality of task processing. After CO, the value drops to nearly 0.7, which eases the computational pressure of the current hot MEC. This will finally reduce the task processing delay and improves the efficiency of task completion.

Figs. 9 and 10 present the computing workload of MECs. From Fig. 9, we observe that the peak value of our proposed method is much lower than the original one, which means computation loads of the hot MEC servers are effectively reduced, hence the task completion delay is reduced as well. Fig. 10 presents the computing workload in different time periods. The number of hot MEC servers at 8:00 is more than that at 20:00, and the heat peak is higher, which accords to the time characteristics of traffic flow. In Shenzhen 8:00 is the rush hour and many people drive out to their working places, while more people choose to stay at home at 20:00. We realize that the heat value of the hot zone drops more at 8:00 than that at 20:00 after CO. In addition, we figure out that the heat peak value of our proposed method decreases more than RTO at 8:00. When heat peak value drops at 20:00, the superior performance of our proposed approach is also reduced. This indicates that the task offloading via our proposed method is able to effectively reduce the excessive workloads especially when the requests of vehicular applications are high.

We then evaluate the energy consumption during the process of MECs cooperation under different MEC layout, i.e., 7×7 and 9×9 distribution of MECs. After the role determination of MECs in these two distributions, Table IV presents the numbers of hot MECs, *non*-hot MECs and neutral MECs at 8:00, 12:00, and 20:00, respectively.

Figs. 11 and 12 present the performance of the cooperative task offloading in terms of energy consumption. The energy saved is represented by the utility function, as seen in (17) and (18). The larger the utility function, the more energy saved, which means lower energy consumption under the MEC distribution. As shown in Fig. 11, the proposed

TABLE IV
NUMBER OF MECS WITH DIFFERENT ROLE AT 8:00, 12:00, AND 20:00

		Hot MEC	*Non*-hot MECs	Neutral MEC
7∗7	8;00	11	29	9
	12:00	11	32	6
	20:00	5	39	5
9∗9	8:00	19	21	41
	12:00	17	28	36
	20:00	6	27	46

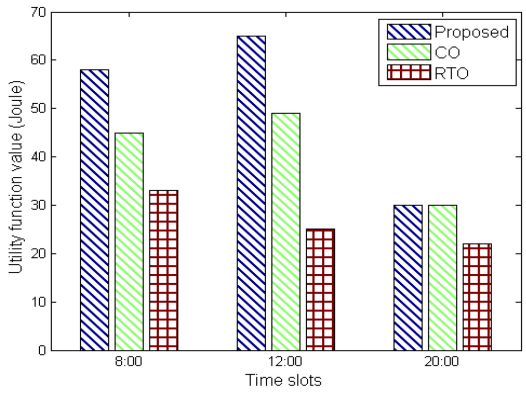


Fig. 11. Energy consumption with 7×7 MECs.

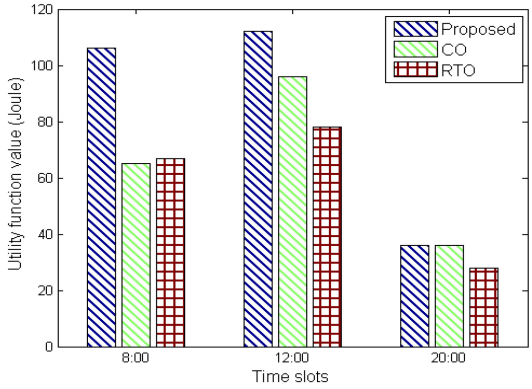


Fig. 12. Energy consumption with 9×9 MECs.

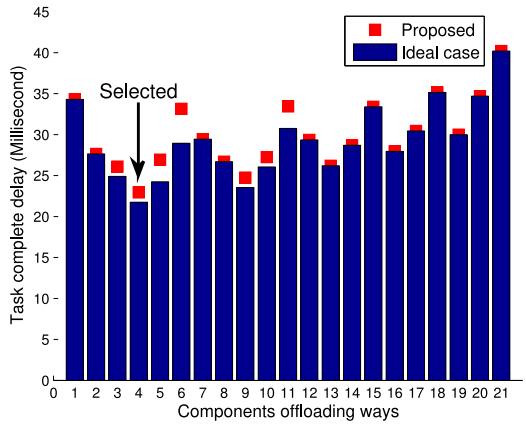


Fig. 13. Task complete delay under different offloading components.

method consumes less energy than RTO and achieves the best performance at 12:00. We observe that the results from Fig. 11 are tightly related to the number of hot and nonhot MEC servers. CO represents a cooperative offloading method without heat prediction, its performance is better than RTO in most cases. In Table IV, we can see that the number of hot MECs at 12:00 is the same as that at 8:00, while there are more nonhot MEC servers at 12:00. As a result, the energy consumption at 12:00 is lower as shown in Fig. 11; however, the nonhot MEC servers at 8:00 are less than that at 20:00 while the hot MEC servers number are more at 8:00, and the saved energy at 8:00 is also more than that at 20:00. In another word, the number of hot MEC servers has greater impact on energy consumption than the nonhot MECs. This can be explained as follows. First, the more hot MECs, the larger strategy set. Second, more nonhot MECs means much more choice of MEC grouping, namely. This is because the greater the number of hot MEC, the more strategy sets, and more strategies are likely to be included in each strategy set. Last but not least, the more strategies, the more space can be chosen, it contributes to find the best strategy and improve the performance of the proposed method. Furthermore, similar findings are supported by the results from Fig. 12. Besides, we find out that the 9×9 MEC layout saves more energy than that the 7×7 does. The reason is that the mutual assistance relationship in the MEC grouping becomes more complicated as the number of MEC servers increases, which actually provides more possibilities to achieve a better strategy.

Fig. 13 presents the comparison of the delay between the real and the ideal situation in the cooperative task offloading. According to Jia *et al.* [47], each task is composed of components with serial dependency. Let (J_0, J) denote different offloading workload. As shown in Fig. 13, taking a task consisting of eight components as an example. Since the first and last components are normally processed locally as *IO* ports [11], we consider components 2–7. Note that continuous components offloading for a task can achieve near-optimal offloading decisions to minimize delay [47]. Thus, according to different offloading components (J_0, J), $J_0 \in \{2,\ldots,7\}$, $J \in \{J_0,\ldots,7\}$, we obtain 21 component series and the corresponding task complete delays are given in Fig. 13. The blue bar represents the ideal case, which assumes the computing resource of all the MECs is sufficient and hence the cooperative task offloading is not required. In Fig. 13, our proposed method adopts to the cooperative task offloading and well deal with the situation of overloaded MEC in hot zones base on minimizing MEC energy consumption. To specify, when the difference between J_0 and J is small, i.e., the offloading workload is small, the task delay of the proposed method is the same with the ideal case. The reason is that less offloading workload means that computing resource of the MEC is sufficient, and there is no need to ask the surrounding MEC servers for help. On the contrary, when the difference becomes larger, the task completion time of the proposed method is higher than the ideal case. The vehicles probably need more than one MEC server to help the task offloading. Under this circumstance, the proposed method can select the optimal strategy thereby being

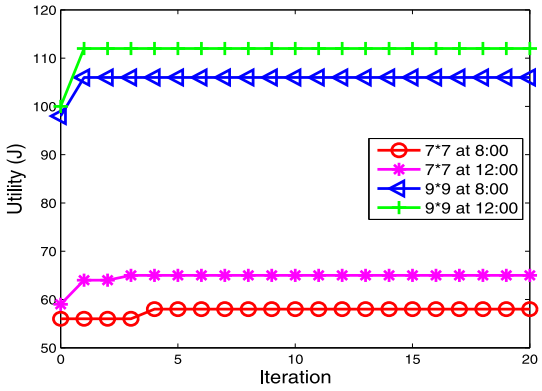


Fig. 14. Convergence of correlated equilibrium.

capable of achieving the minimum delay due to task computing hungry. This indicates that the transmission and queuing delays caused by CO are smaller than the reduction in computing CO. Thus, the final "Selected" component is the way that indeed minimizes the task complete delay. Combining the results in Fig. 13 with the ones described in Figs. 11 and 12, we conclude that our method is inherently able to obtain a good tradeoff in reducing the latency and energy consumption.

Furthermore, we evaluate the convergence performance of the proposed noncooperative game-theoretical approach. As shown in Fig. 14, we consider the 9×9 and 7×7 layout in the selected area at 8:00 and 12:00, respectively. The results demonstrate that the proposed method can achieve correlated equilibrium with only a few iterations, hence it is capable of satisfying the requirement of high mobility in the vehicular application scenarios. Moreover, we observe that the utility of the 9×9 layout is larger than that in 7×7. The reason behind this is that the 9×9 MEC layout has more complicated cooperation relationship than the 7×7, and the proposed method can save more energy in 9×9 then the utility is higher. In other words, our proposed method is able to work properly as the number of related MECs increases in the MEC cooperation.

VII. Conclusion

In this article, we strived to offer vehicular task offloading via heat-aware MEC cooperation with a noncooperative game theory approach. To specify, we first established the association between the unique stop-and-wait behavior and the heat of hot zones, and designed a GRU-based prediction method, guiding the MECs role determination through the analysis of real-world large-scale private car trajectory data set. Then we formulated a noncooperative game-theoretic strategy selection to solve the MEC grouping problem, with which the MECs obtained the optimal strategy based on the regret matching, with the aim at maximizing the overall utility and achieving the correlated equilibrium. The simulation results demonstrate that the proposed method significantly reduces the task complete delay, as well as improves the MEC energy efficiency with end users' QoE guaranteed. In the future, we will devote our effort to combining cooperative task offloading with the spectrum resource allocation among MECs more tightly, and consider more complicated models to describe the relationship among components in our future work.

References

[1] Y. Ni, J. He, L. Cai, J. Pan, and Y. Bo, "Joint roadside unit deployment and service task assignment for Internet of Vehicles (IoV)," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3271–3283, Apr. 2019.

[2] Z. Xiao *et al.*, "Spectrum resource sharing in heterogeneous vehicular networks: A noncooperative game-theoretic approach with correlated equilibrium," *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9449–9458, Oct. 2018.

[3] H. Marzbani, H. Khayyam, C. N. TO, D. V. Quoc, and R. N. Jazar, "Autonomous vehicles: Autodriver algorithm and vehicle dynamics," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3201–3211, Apr. 2019.

[4] X. Cheng, C.-X. Wang, B. Ai, and H. Aggoune, "Envelope level crossing rate and average fade duration of nonisotropic vehicle-to-vehicle Ricean fading channels," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 62–72, Feb. 2014.

[5] J. Liu, W. Wang, D. Li, S. Wan, and H. Liu, "Role of gifts in decision making: An endowment effect incentive mechanism for offloading in the IoV," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 6933–6951, Aug. 2019.

[6] Y. Jiang and D. H. K. Tsang, "Delay-aware task offloading in shared fog networks," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4945–4956, Dec. 2018.

[7] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.

[8] H. Guo and J. Liu, "Collaborative computation offloading for multiaccess edge computing over fiber-wireless networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 4514–4526, May 2018.

[9] Y. Sun *et al.*, "Adaptive learning-based task offloading for vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3061–3074, Apr. 2019.

[10] S. E. Mahmoodi, R. N. Uma, and K. P. Subbalakshmi, "Optimal joint scheduling and cloud offloading for mobile applications," *IEEE Trans. Cloud Comput.*, vol. 7, no. 2, pp. 301–313, Apr. 2019.

[11] S. Yu, R. Langar, X. Fu, L. Wang, and Z. Han, "Computation offloading with data caching enhancement for mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11098–11112, Nov. 2018.

[12] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.

[13] L. Chen, S. Zhou, and J. Xu, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1619–1632, Aug. 2018.

[14] R.-H. Huang, B.-J. Chang, Y.-L. Tsai, and Y.-H. Liang, "Mobile edge computing-based vehicular cloud of cooperative adaptive driving for platooning autonomous self driving," in *Proc. IEEE 7th Int. Symp. Cloud Service Comput. (SC2)*, Nov. 2017, pp. 32–39.

[15] A. L. Alfeo, M. G. C. A. Cimino, S. Egidi, B. Lepri, and G. Vaglini, "A Stigmergy-based analysis of city hotspots to discover trends and anomalies in urban transportation usage," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2258–2267, Jul. 2018.

[16] 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.

[17] W. Fan, Y. Liu, B. Tang, F. Wu, and Z. Wang, "Computation offloading based on cooperations of mobile edge computing-enabled base stations," *IEEE Access*, vol. 6, pp. 22622–22633, 2018.

[18] D. Wang *et al.*, "Stop-and-wait: Discover aggregation effect based on private car trajectory data," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3623–3633, Oct. 2019.

[19] P. Si, Y. He, H. Yao, R. Yang, and Y. Zhang, "DaVe: Offloading delay-tolerant data traffic to connected vehicle networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3941–3953, Jun. 2016.

[20] Z. Xiao *et al.*, "Towards accurate vehicle state estimation under non-Gaussian noises," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10652–10664, Dec. 2019.

[21] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[22] M. Peng, S. Yan, K. Zhang, and C. Wang, "Fog-computing-based radio access networks: Issues and challenges," *IEEE Netw.*, vol. 30, no. 4, pp. 46–53, Jul./Aug. 2016.

[23] T. Dang and M. Peng, "Joint radio communication, caching, and computing design for mobile virtual reality delivery in fog radio access networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 7, pp. 1594–1607, Jul. 2019.
[24] Z. Zhao *et al.*, "On the design of computation offloading in fog radio access networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 7136–7149, Jul. 2019.
[25] M. Peng and K. Zhang, "Recent advances in fog radio access networks: Performance analysis and radio resource allocation," *IEEE Access*, vol. 4, pp. 5003–5009, 2016.
[26] H. Xiang, W. Zhou, M. Daneshmand, and M. Peng, "Network slicing in fog radio access networks: Issues and challenges," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 110–116, Dec. 2017.
[27] Y. Zhang, J. He, and S. Guo, "Energy-efficient dynamic task offloading for energy harvesting mobile cloud computing," in *Proc. IEEE Int. Conf. Netw. Architect. Storage (NAS)*, Oct. 2018, pp. 1–4.
[28] Y. Kim, J. Kwak, and S. Chong, "Dual-side optimization for cost-delay tradeoff in mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1765–1781, Feb. 2018.
[29] J. L. D. Neto, S. Yu, D. F. Macedo, J. M. S. Nogueira, R. Langar, and S. Secci, "ULOOF: A user level online offloading framework for mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2660–2674, Nov. 2018.
[30] Y. Wang, K. Wang, H. Huang, T. Miyazaki, and S. Guo, "Traffic and computation co-offloading with reinforcement learning in fog computing for industrial applications," *IEEE Trans. Ind. Informat.*, vol. 15, no. 2, pp. 976–986, Feb. 2019.
[31] L. Cui *et al.*, "Joint optimization of energy consumption and latency in mobile edge computing for Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4791–4803, Jun. 2019.
[32] X. Ge, "Ultra-reliable low-latency communications in autonomous vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5005–5016, May 2019.
[33] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu, "A scalable and quick-response software defined vehicular network assisted by mobile edge computing," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 94–100, Jul. 2017.
[34] H. Zhou *et al.*, "WhiteFi infostation: Engineering vehicular media streaming with geolocation database," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 8, pp. 2260–2274, Aug. 2016.
[35] J. Du, F. R. Yu, X. Chu, J. Feng, and G. Lu, "Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1079–1092, Feb. 2019.
[36] J. Guo, B. Song, F. R. Yu, Y. Chi, and C. Yuen, "Fast video frame correlation analysis for vehicular networks by using CVS–CNN," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6286–6292, Jul. 2019.
[37] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of Vehicles: A fog-enabled real-time traffic management system," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4568–4578, Oct. 2018.
[38] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[39] Y. Hao, M. Chen, L. Hu, M. S. Hossain, and A. Ghoneim, "Energy efficient task caching and offloading for mobile edge computing," *IEEE Access*, vol. 6, pp. 11365–11373, 2018.
[40] L. Cheng and T. Yu, "Game-theoretic approaches applied to transactions in the open and ever-growing electricity markets from the perspective of power demand response: An overview," *IEEE Access*, vol. 7, pp. 25727–25762, 2019.
[41] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
[42] M. Liwang, J. Wang, Z. Gao, X. Du, and M. Guizani, "Game theory based opportunistic computation offloading in cloud-enabled IoV," *IEEE Access*, vol. 7, pp. 32551–32561, 2019.
[43] S. Hart, S. Li, and A. Mas-Colelil, "A simple adaptive procedure leading to correlated equilibrium," *Econometric*, vol. 68, no. 5, pp. 1127–1150, Sep. 2000.
[44] J. W. Huang and V. Krishnamurthy, "Cognitive base stations in LTE/3GPP femtocells: A correlated equilibrium game-theoretic approach," *IEEE Trans. Commun.*, vol. 59, no. 12, pp. 3485–3493, Dec. 2011.
[45] D. Nowak, T. Mahn, H. Al-Shatri, A. Schwartz, and A. Klein, "A generalized nash game for mobile edge computation offloading," in *Proc. 6th IEEE Int. Conf. Mobile Cloud Comput. Services Eng. (MobileCloud)*, Mar. 2018, pp. 95–102.
[46] L. Li, H. Zhou, S. X. Xiong, J. Yang, and Y. Mao, "Compound model of task arrivals and load-aware offloading for vehicular mobile edge computing networks," *IEEE Access*, vol. 7, pp. 26631–26640, 2019.
[47] M. Jia, J. Cao, and L. Yang, "Heuristic offloading of concurrent tasks for computation-intensive applications in mobile cloud computing," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2014, pp. 352–357.
[48] A. S. Prasetia, R. Wai, Y. Wen, and Y. Wang, "Mission-based energy consumption prediction of multirotor UAV," *IEEE Access*, vol. 7, pp. 33055–33063, 2019.
[49] X. Ding and J. Wu, "Study on energy consumption optimization scheduling for Internet of Things," *IEEE Access*, vol. 7, pp. 70574–70583, 2019.
[50] O. A. Alimi and K. Ouahada, "Smart home appliances scheduling to manage energy usage," in *Proc. IEEE 7th Int. Conf. Adapt. Sci. Technol. (ICAST)*, Aug. 2018, pp. 1–5.
[51] X. Feng, J. Wang, X. Kong, Z. Wang, and C. Liu, "Exploring human mobility patterns in urban scenarios: A trajectory data perspective," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 142–149, Jul. 2018.
[52] Y. Huang, Z. Xiao, X. Yu, D. Wang, V. Havyarimana, and J. Bai, "Road network construction with complex intersections based on sparsely-sampled private car trajectory data," *ACM Trans. Knowl. Disc. Data*, vol. 13, no. 35, pp. 1–28, Jul. 2019.
[53] Y. Deng, L. Wang, H. Jia, X. Tong, and F. Li, "A sequence-to-sequence deep learning architecture based on bidirectional GRU for type recognition and time location of combined power quality disturbance," *IEEE Trans. Ind. Informat.*, vol. 15, no. 8, pp. 4481–4493, Aug. 2019.
[54] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018.

Zhu Xiao (Senior Member, IEEE) received the M.S. and Ph.D. degrees in communication and information system from Xidian University, Xi'an, China, in 2007 and 2009, respectively.

From 2010 to 2012, he was a Research Fellow with the Department of Computer Science and Technology, University of Bedfordshire, Luton, U.K. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. His research interests include mobile communications, wireless localization, Internet of Vehicles, and mobile computing.

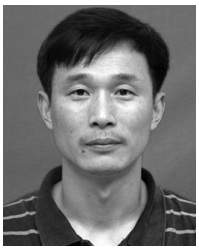
Xingxia Dai received the B.S. degree from Xiangtan University, Xiangtan, China, in 2018. She is currently pursuing the M.S. degree in communication and information system with Hunan University, Changsha, China.

Her interests include Internet of Vehicles and mobile edge computing.

Hongbo Jiang (Senior Member, IEEE) received the Ph.D. degree from Case Western Reserve University, Cleveland, Ohio, in 2008.

He is currently a Full Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. He ever was a Professor with the Huazhong University of Science and Technology, Wuhan, China. His research concerns computer networking, especially algorithms and protocols for wireless, and mobile networks.

Prof. Jiang is serving as an Editor for the IEEE/ACM TRANSACTIONS ON NETWORKING, an Associate Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING, and an Associate Technical Editor for *IEEE Communications Magazine*.



Dong Wang (Member, IEEE) received the B.S. degree and the Ph.D. degree in computer science from Hunan University, Changsha, China, in 1986 and 2006, respectively.

From December 2004 to December 2005, he was a Visiting Scholar with the University of Technology Sydney, Ultimo, NSW, Australia. Since 1986, he has been working with Hunan University, where he is currently a Professor. His main research interests include network test and performance evaluation, wireless communications, and mobile computing.

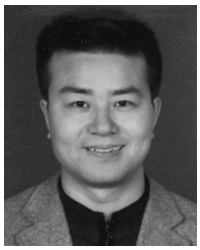
Hongyang Chen (Senior Member, IEEE) received the Ph.D. degree from the University of Tokyo, Tokyo, Japan.

In 2009, he was a Visiting Researcher with the UCLA Adaptive Systems Laboratory, University of California at Los Angeles, Los Angeles, CA, USA. His research interests include IoT, intelligent networking, location estimation and location based service, and signal processing for communications.

Dr. Chen served as the Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, and the Associate Editor for *IEEE Communications Letters*.

Liang Yang (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from Sun Yat-sen University, Guangzhou, China, in 2006.

From 2006 to 2013, he was a faculty with Jinan University, Guangzhou. He joined the Guangdong University of Technology, Guangzhou, in 2013. He is currently a Professor with Hunan University, Changsha, China. His current research interests include the performance analysis of wireless communications systems.



Fanzi Zeng (Member, IEEE) received the Ph.D. degree in signal and information processing from Beijing Jiao Tong University, Beijing, China, in 2005.

Since 2005, he has been with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China, where he is currently a Professor. In 2009, he was with the Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI, USA, as a Visiting Scholar. His current research focuses on cognitive radio technology and artificial intelligence.