IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON EMERGING TECHNOLOGIES ON VEHICLE TO EVERYTHING (V2X)

Received December 17, 2018, accepted January 6, 2019, date of publication February 22, 2019, date of current version March 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2896000

Matching-Based Task Offloading for Vehicular Edge Computing

PENGJU LIU, JUNLUO LI, AND ZHONGWEI SUN

State Key Laboratory of Alternate Electrical Power System With Renewable Energy Sources, School of Electrical and Electronic Engineering, North China Electric Power University, Beijing, China

Corresponding author: Zhongwei Sun (zwsun@ncepu.edu.cn)

ABSTRACT Vehicular edge computing has emerged as a promising technology to accommodate the tremendous demand for data storage and computational resources in vehicular networks. By processing the massive workload tasks in the proximity of vehicles, the quality of service can be guaranteed. However, how to determine the task offloading strategy under various constraints of resource and delay is still an open issue. In this paper, we study the task offloading problem from a matching perspective and aim to optimize the total network delay. The task offloading delay model is derived based on three different velocity models, i.e., a constant velocity model, vehicle-following model, and traveling-time statistical model. Next, we propose a pricing-based one-to-one matching algorithm and pricing-based one-to-many matching algorithms for the task offloading. The proposed algorithm is validated based on three different simulation scenarios, i.e., straight road, the urban road with the traffic light, and crooked road, which are extracted from the realistic road topologies in Beijing and Guangdong, China. The simulation results confirm that significant delay decreasing can be achieved by the proposed algorithm.

INDEX TERMS Vehicular edge computing, task offloading, one-to-one matching, matching with quota, SUMO.

I. INTRODUCTION

With the explosive development of wireless communications and Internet of vehicles, the amount of intelligent transportation applications such as automatic driving and vehicular video streaming has been increasing consistently. The successful implementation of these emergent applications requires processing a large number of tasks with high computational complexity and strict delay sensitivity [1]. Due to the limited processing capability of vehicles, tasks have to be offloaded from vehicles to remote cloud servers via cellular networks [2]–[4]. However, this not only puts a heavy burden on the already congested cellular networks, but also causes a high computational delay due to the long distance between vehicles and the cloud [5], [6]. To address this challenge, vehicular edge computing (VEC) [7], which combines edge computation and vehicle networks, has emerged as a promising solution.

Processing the task at the network edge has the following advantages [8]. First, it can relieve the network overload since the large amount of data needs not to travel through the whole network. Besides, it can avoid migrating the task in the duplicated way, so the service quality can be enhanced efficiently [9]. Second, due to the proximity between vehicles and servers, the computational delay can be reduced. Last but not least, it can increase resource utilization efficiency by leveraging distributed servers with under-utilized resources, and effectively process multi-source heterogeneous data [10].

Despite these advantages, VEC also meets some challenges [11]. First, the vehicle mobility has a large impact on the task offloading optimization. A vehicle can only communicate with a road side unit (RSU) and upload its task when it is within the coverage [12], [13]. The transmission process will be interrupted once the vehicle moves out of the coverage. Therefore, a precise estimation of the vehicle dwell time is necessary for the optimization of task offloading. Second, the self-interested and rational vehicles have their own preferences towards edge servers. It is difficult to derive a unified task offloading decision which can meet the interests of each vehicle [14]. Third, most of the conventional task offloading schemes are derived and evaluated based on theoretical models. There lacks a comprehensive evaluation under realistic traffic data to simulate a dynamic environment and reflect the true vehicular performance.

The associate editor coordinating the review of this manuscript and approving it for publication was Di Zhang.

In this paper, we propose a task offloading algorithm based on matching theory. The vehicles are considered as the one side of matching, and the road side units (RSU) are considered as the other [15]. The proposed algorithm aims at minimizing the total offloading delay. Besides, three vehicular mobility models, i.e., constant velocity model, vehicle-following model, and travelling-time model, are proposed to estimate the delay, and the proposed algorithm is evaluated under three realistic scenarios, i.e., straight road scenario, urban road scenario with traffic light, and crooked road scenario. The main contributions of this paper are summarized as follows.

- **Various mobility and delay models:** Considering the mobility of vehicles and the dynamic environment, we study three vehicular mobility models to simulate the movement of vehicles and derive the explicit expression of waiting delay model and handover delay model.
- **Matching-based task offloading:** We propose a matching-based task offloading algorithm to minimize the network delay, in which both the one-to-one and one-to-many matching are considered.
- **Performance evaluation under various real-world scenarios:** The proposed algorithm is evaluated under three realistic roads, i.e., straight road, urban road with traffic light, and crooked road. The corresponding data are obtained from SUMO and then used for the simulation of task offloading in MATLAB.

The remaining parts of this paper are organized as follows. The related works are presented in Section II. The system model is introduced in Section III. Section IV describes the problem formulation of the task offloading problem. Section V and Section VI propose the task offloading based on one-to-one matching algorithm and matching with quota algorithm, respectively. Numerical Results are provided in Section VII. Section VIII concludes this paper.

II. RELATED WORKS

The objective of this work is to study the task offloading problem in VEC. Edge computing has attracted intensive research from both academia and industry [16]. Abbas *et al.* [17] provided a mobile edge computing architecture and elaborated its advantages, potential application areas and future research directions. Rahman *et al.* [18] developed a mobile edge computing framework to support real-time and location-aware personalized services. Baktir *et al.* [19] provided a comprehensive survey of edge computing and discussed its technical challenges in depth.

In mobile edge computing, one of the most important problems is task offloading [20]. A set of studies have already focused on how to optimize task offloading from different perspectives. Liu *et al.* [21] proposed a distributed computation offloading algorithm based on game theory. Ali *et al.* [22] developed a distributed and self-organizing method to solve the matching game to minimize the end-to-end latency in Internet of Things (IoT) networks. Gu *et al.* [23] designed a task assignment mechanism to reduce overall energy consumption, which can also satisfy the heterogeneous delay requirements and support good scalability. However, these works have not considered the mobility of vehicles and the complicated transportation scenarios.

There exist some works which have investigated task offloading in VEC. Qiao *et al.* [24] developed a novel paradigm to offload the computational-intensive tasks to heterogeneous mobile edge computing servers and resource-rich vehicles. In [21], in order to reduce the latency of the computation offloading of vehicles, Liu *et al.* formulate the problem as a multi-user task offloading problem, and proposed a distributed task offloading algorithm to reduce the offloading delay of vehicles. Zhou *et al.* [25] proposed an energy-efficient VEC framework for in-vehicle user equipment (UEs) with limited battery capacity, and developed an alternating direction method of multipliers (ADMM)-based energy-efficient resource allocation algorithm. Liu and Zhang [14]investigated the task offloading problem, and proposed a heuristic searching algorithm to solve the problem by optimizing candidates selection, offloading ordering and task allocation.

Matching theory provides a strong tool to solve the combinational problem about task offloading [26]. It can be used to study the establishment of dynamic and mutually beneficial relations. It is particularly effective in developing extendable, flexible, decentralized, and practical solutions for some complex networks [27]. In particular, it can effectively deal with the high dynamics of networks, by considering the competitive, distributed nature of network elements, limited radio resources, and the dynamic quality of service (QoS) constraints of different elements [28], [29]. The matching theory originates from stable marriage problem (SM). There are some classical algorithms such as the conventional Gale-Shapley algorithm [30], swap matching algorithm [31], and pricing-based matching algorithm [27] to solve the general concepts of matching models. Matching problem can also be divided into four categories, i.e., one-to-one, one-to-many, many-to-one, and many-to-many matching. Wang *et al.* [32] proposed a one-to-one stable matching algorithm for latency optimization in the D2D-based social IoT networks. Zhao *et al.* [33] proposed a novel algorithm for obtaining a sub-optimal solution based on the many-to-many two-sided matching game with externalities. Gu *et al.* [34] introduced the idea of cheating in matching to further improve the throughput of D2D communications.

We also resort matching to solve the vehicle task offloading problem. Our feature is that we employ different velocity models to analyze the complex delay models. Particularly, we study three vehicular mobility models and derive the mathematic expressions of constant velocity model, vehicle-following model and travelling-time statistical model. Furthermore, we also use SUMO to evaluate our algorithm based on realistic road topologies [35] because a realistic evaluation scenario is crucial to evaluate the performance of proposed algorithm. SUMO is a road traffic simulation software to evaluate the real-world road topologies without deploying

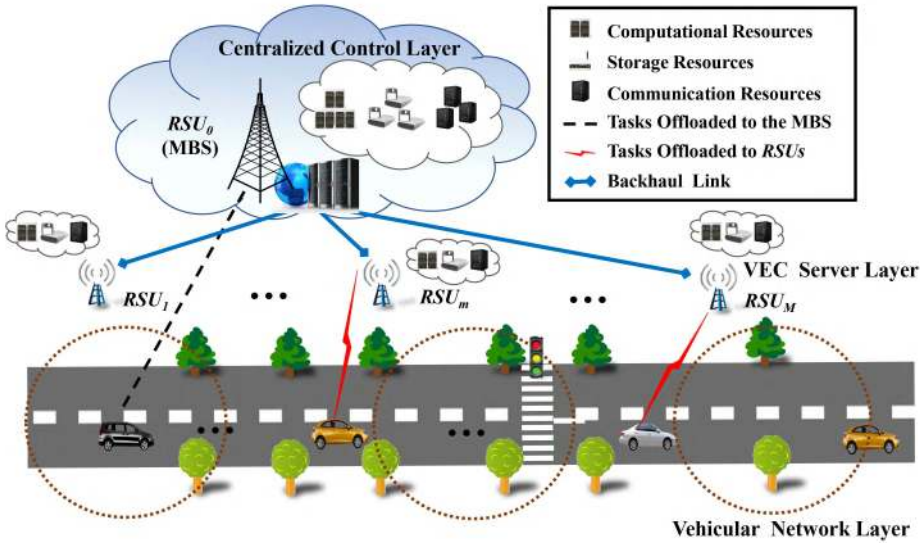


FIGURE 1. The architecture of VEC.

a physical equipment to catch realistic data and has been widely used for performance evaluation in vehicular networks. In [36], the proposed content distribution algorithm was compared with two heuristic schemes based on real-world map and realistic vehicular traffic by employing both SUMO and MATLAB. Zhou *et al.* [37], investigated the content distribution problem in D2D-based cooperative vehicular networks, based on realistic vehicular traffic provided by SUMO, which is useful to give a more precise construction of a dynamic environment. Codeca *et al.* [36] used SUMO to rebuild a realistic traffic pattern and dealt with traffic congestion problems.

III. SYSTEM MODEL

The three-layer VEC framework is presented in Fig.1, which is composed of the centralized control layer, the distributed VEC layer and the vehicular network layer.

The centralized control layer is responsible for task assignment and handover management. The macro base station (MBS) locates at the center of the road, and we assume that the coverage of the MBS is large enough to ensure that all the vehicles can access to it. The MBS connects with a remote cloud server, which can offer the computational resource to the MBS. Here, we consider that the remote cloud server has enough computational resource but causes a heavy delay due to the long distance. The MBS can determine the task offloading strategy based on the collected information from vehicles (the task computing requirements, position, and velocity information, etc.), RSUs and the remote cloud server (the idle computational resources and channel state information, etc.).

In the distributed VEC layer, M RSUs with different coverage areas are considered to be deployed along this unidirectional road, the set of which is denoted as $\mathcal{RSU} = \{RSU_1, \cdots, RSU_m, \cdots, RSU_M\}$, and the corresponding set of indices is denoted as $\mathcal{M} = \{1, \cdots, m, \cdots, M\}$. The coverage radius of RSU_m is denoted as r_m. Based on the coverage area of M RSUs with different radius, the road is divided into M segments with different sizes. A vehicle can only communicate with RSU_m when it is located in the section m. For any $RSU_m \in \mathcal{RSU}$, it has a co-located edge server to offer the computational resource, which is denoted as δ_m. For the sake of narrative, the MBS can be denoted as RSU_0 and added into the set $\mathcal{RSU}$. Besides, the corresponding index is $m = 0$ and the computational resource of the remote cloud server is denoted as δ_0.

The vehicular network layer is composed of N vehicles travelling towards the same direction into an unidirectional road. Denote the set of vehicles as $\mathcal{V} = \{V_1, \cdots, V_n, \cdots, V_N\}$, and the corresponding set of indices is denoted as $\mathcal{N} = \{1, \cdots, n, \cdots, N\}$. The velocity and acceleration of V_n are denoted as v_n and a_n, respectively. For each vehicle V_n, it will generate a task when it arrives at the road, which can be characterized by a triplet $\{D_n, C_n, \tau_n\}$. D_n represents the data size of the task (bits), C_n denotes the computational resource demand for processing the task (MHz), and τ_n is the delay constraint (seconds).

The task offloading and execution process are implemented as follows. Firstly, each vehicle $V_n \in \mathcal{V}$ informs the MBS of its task computational requirement, i.e., D_n, C_n, and τ_n. Then, the MBS determines to offload the tasks either to the RSUs or the MBS for execution in order to reduce the offloading delay. Next, the MBS will inform the vehicles with the task offloading decision, based on which, the task will be either offloaded to the corresponding RSU, or to the MBS. If the vehicle chooses the RSU to offload its task, the task will be processed by the edge server with a short delay. Otherwise, the task will be offloaded to the MBS and processed by

TABLE 1. Parameter.

Variable	Notation
M	The number of RSU
N	The number of vehicle
r_m	The radius of RSU_m
δ_m	CPU cycle
v_n	The velocity of V_n
a_n, a_n'	The acceleration of V_n
D_n	Data size of task
C_n	Computation resource demand
τ_n	Delay constant
$S_m, d_n^a, d_n^{a'}, d_n^v, d_n$	Travelling distance of vehicle
$t_n^a, t_n^{a'}, t_n^v, t_n$	Travelling time of vehicle
K, K'	The number of traffic lights
θ	Scale parameter
$T_{n,m}^w$	Waiting delay
$T_{n,m}^t$	Transmission delay
$T_{n,m}^c$	Computational delay
$T_{m,m'}^h$	Handover delay
$T_{n,m}$	Task offloading delay
T_cloud	Additional delay in cloud computing
$\gamma_{n,m}$	SNR between V_n and RSU_m
P_t	Transmission power
$g_{n,m}$	Channel power gain
N_0	Power spectral density of AWGN
B	Channel bandwidth
$R_{n,m}$	Data transmission rate

the remote cloud server with a heavy delay, though it has ample computational resource. During the task computational period, a vehicle may travel into the coverage of another RSU due to the mobility of vehicles, in this situation, the MBS has to collect all the task computational results. When the task has been processed, the computational result will be feed back to the RSU, in which the vehicle is located currently, then the result is transmitted to the target vehicle.

Denote the total offloading delay as the time difference between the vehicle entering the road and receiving the computational results, which consists of transmission delay, task computational delay, waiting delay, and handover delay. Among the various types of delay, the waiting delay and handover delay depend on the specific mobility models. In this paper, we consider three types of mobility models to simulate the motion of vehicles and calculate the waiting delay and handover delay, which are constant velocity model, vehicle-following model and travelling-time statistical model. Besides, we compare the performance of these three models with the real motion data generated by SUMO. The detailed mobility model, handover model, and delay model are elaborated as follows. The mathematical variables used throughout this work are summarized in Table 1.

A. THE MOBILITY MODEL AND WAITING DELAY

If the vehicle V_n determines to offload its tasks to RSU_m, a precedent condition is that the vehicle must locate in the coverage of RSU_m. That is to say, the vehicle V_n cannot offload its task to RSU_m until it reaches the coverage of RSU_m, and the corresponding waiting time is called waiting delay, which is denoted as $T_{n,m}^w$. In this section, we introduce three vehicular mobility models, and derive the corresponding waiting delay.

Without loss of generality, we consider a scenario that the RSUs are arrayed from left side of the road to right side, i.e., RSU_1 is at the far left of the road and RSU_M is at the far right of the road, as shown in Fig. 1. Furthermore, we assume that all vehicles enter into the road from the left edge of the road, i.e., the coverage of the RSU_1. Thus, the travelling distance of any vehicle until it enters into the coverage of RSU_m can be calculated as

$$S_m = 2\sum_{i=1}^{m-1} r_i. \tag{1}$$

Remark 1: If the vehicle V_n determines to offload its tasks to the MBS or RSU_1, there will be no travelling distance and waiting delay, i.e., the travelling distance is $S_0 = S_1 = 0$, and the waiting delay is $T_{n,0}^w = T_{n,1}^w = 0$.

Then, vehicular mobility models are described in details as follows.

1) CONSTANT VELOCITY MODEL

We assume that any vehicle $V_n \in \mathcal{V}$ travels with a constant velocity v_n when it enters into the road, and the velocity of each vehicle V_n is uniformly distributed in the range of [0, 27.7] m/s. Then the waiting delay of the vehicle V_n can be calculated as

$$T_{n,m}^w = \frac{S_m}{v_n}. \tag{2}$$

2) VEHICLE-FOLLOWING MODEL

In the urban traffic scenario, the vehicle-following model is often adopted, in which the vehicle velocity is affected by numerous factors [38], such as the traffic lights, the mobile behavior of the leading vehicle, etc. We mainly consider the impact of traffic lights here and the impact of the leading vehicle can be analyzed similarly. Specifically, we assume that there exists K traffic lights located evenly in this road.

The vehicle-following model consists of three phases when the vehicle V_n passes a traffic light, which are the continuous braking phase, the stationary phase, and the acceleration phase. The details are described as follows.

The vehicle V_n is assumed to travel with a constant velocity v_n initially. During the continuous braking phase, the velocity of the vehicle V_n will decrease from v_n to zero with a constant deceleration a_n after the driver notices the traffic light and takes the braking action. The distance of the vehicle V_n in this stage is calculated as

$$d_n^a = \frac{v_n^2}{2a_n}, \tag{3}$$

and the braking time t_n^a is calculated as

$$t_n^a = \frac{v_n}{a_n}. \tag{4}$$

Then during the stationary phase, the vehicle V_n will wait for a period of time t_n^w at the intersection, which is

$$t_n^w = \beta_n T_{max}, \tag{5}$$

where T_{max} is the predetermined maximum waiting delay for red light. β_n is the ratio of the remaining waiting delay for red light to the maximum waiting delay T_{max}, which is related to the time when the vehicle arrives at the intersection and can be seen as a random value in the simulation.

During the acceleration phase, the velocity of the vehicle V_n will increase from zero to v_n again with a constant acceleration a'_n. Similar to those in the continuous braking phase, the vehicle V_n travelling distance $d_n^{a'}$ and the corresponding acceleration time $t_n^{a'}$ in this phase can be expressed as

$$d_n^{a'} = \frac{v_n^2}{2a'_n}, \tag{6}$$

$$t_n^{a'} = \frac{v_n}{a'_n}. \tag{7}$$

As discussed above, the total travelling distance of the vehicle V_n passing an intersection with the traffic light can be calculated by

$$d_n^v = d_n^a + d_n^{a'}, \tag{8}$$

and the total time of the vehicle V_n passing an intersection is

$$t_n^v = t_n^a + t_n^w + t_n^{a'}. \tag{9}$$

Considering there are K traffic lights located in the road, and the vehicle V_n has passed by K'_n traffic lights before it enters to the segment m. Then, the waiting delay for the vehicle V_n entering the coverage of RSU_m is composed of two part. One is the uniform motion phase and the moving distance can be expressed as the difference between S_m and $K'_n d_n^v$. The other is the continuous braking phase and acceleration phase, the total travelling time can be expressed as $K'_n t_n^v$. From what has been discussed above, the waiting delay can be obtained as

$$T_{n,m}^w = \frac{S_m - K'_n d_n^v}{v_n} + K'_n t_n^v, \tag{10}$$

here, we must have $K'_n \leq K$.

3) TRAVELLING-TIME STATISTICAL MODEL

The two theoretical models described above are too optimistic for real-world implementation, especially in complex traffic scenarios. Several works have focused on the vehicular travelling time for a given road segment, i.e., the waiting delay, which is demonstrated to follow a Gamma distribution [39], [40]. The details are described as follows.

Generally, if the vehicle V_n travels for a distance d_n, and the corresponding time is denoted as t_n, then the probability distribution function (PDF) of Gamma distribution $Ga(d_n, \theta)$ can be expressed as

$$f(t_n, d_n, \theta) = \frac{t_n^{d_n - 1} e^{-\frac{t_n}{\theta}}}{\theta^{d_n} \Gamma(d_n)}, (d_n > 0, \theta > 0), \tag{11}$$

where d_n is also called the shape parameter and θ is called the scale parameter. $\Gamma(d_n)$ is the Gamma function which is given by

$$\Gamma(d_n) = \int_0^{+\infty} t_n^{d_n - 1} e^{-t_n} dt_n. \tag{12}$$

Therefore, when the travelling distance of the vehicle V_n is S_m, i.e., $d_n = S_m$, the waiting delay is assumed to follow a Gamma distribution, i.e., $t_n \sim Ga(S_m, \theta)$, where θ is determined by the actual conditions of the road segment [41] and can be obtained based on the historical traffic information, including the vehicle trajectory, road congestion, accident probability and so on. We take the expectation of t_n as the waiting delay of the vehicle V_n arriving at the coverage of RSU_m, i.e.,

$$T_{n,m}^w = S_m \theta. \tag{13}$$

4) MOBILITY MODEL BY SUMO

In addition to the three vehicular mobility models described above, we can get the actual motion data of the vehicle by SUMO.

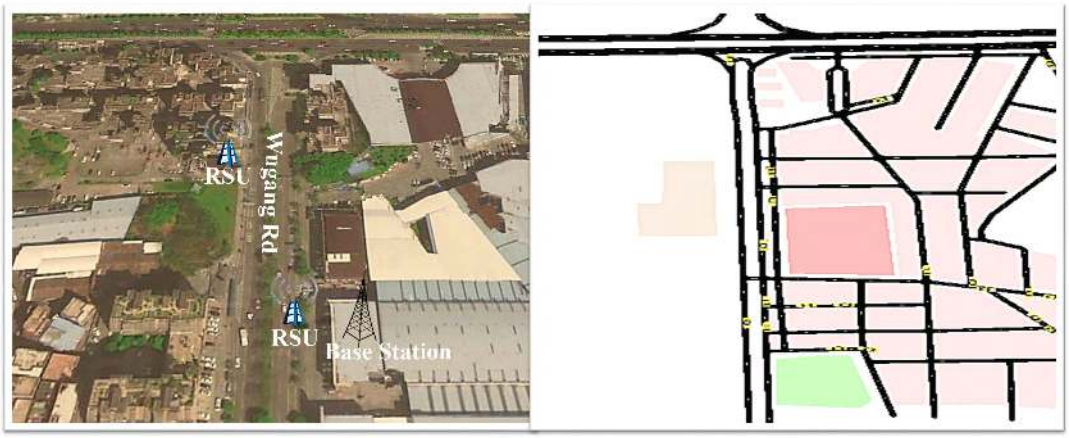
The traffic simulator software, SUMO, is able to select the real-world map for simulation, and with which we can avoid the expensive costs for deploying a physical transportation measurement system. Each vehicle V_n is treated as an independent element in SUMO and various mobility parameters of vehicles such as acceleration, deceleration, velocity, and route, can be adjusted and controlled separately. Particularly, we consider three different scenarios, which are the straight road, urban road with traffic light and crooked road, as shown in Fig. 2,3 and 4, respectively. The application of TSUMO with different scenarios is described in details in Section VII.

B. THE DATA TRANSMISSION MODEL

Assuming that each vehicle V_n is allocated with orthogonal channel, i.e., there is no interference among the vehicles. Considering the data are transmitted from the vehicle V_n to RSU_m, the effective signal to noise ratio (SNR) of the transmission link between V_n and RSU_m can be expressed as

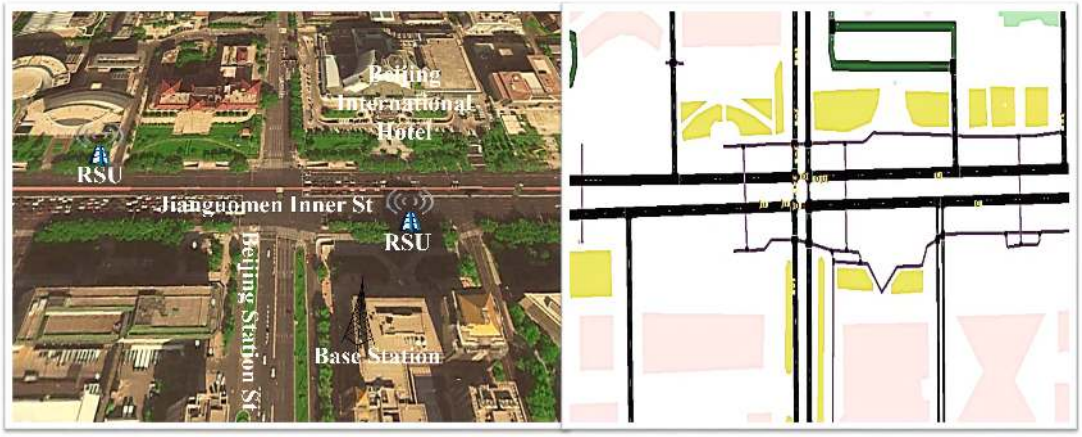
$$\gamma_{n,m} = \frac{P_t g_{n,m}}{N_0}, \tag{14}$$

where P_t is the transmission power of the vehicle V_n, which is constant for each vehicle V_n. N_0 is the power of the additive white Gaussian noise (AWGN). $g_{n,m}$ represents the channel power gain of the transmission link from the vehicle V_n to RSU_m. Due to the mobility of vehicle, the channel varies rapidly and it is difficult to get the real-time channel state information. Previous works have verified that only considering the large-scale fading causes little performance degradation. For the sake of simplification, we ignore the small-scale fading and the channel power gain is expressed as $g_{n,m} = r_m^{-\alpha}$ [21]. Here, we consider the vehicle V_n transmits its task once it enters to the coverage of RSU_m, and α is pathloss exponent. Then the data transmission rate of the



(a) The real-world map of Wugang Road in Foshan, China. (b) The simulation scenario based on SUMO.

FIGURE 2. The scenario of straight road.



(a) The real-world map of Jianguomen Overpass in Beijing, China. (b) The simulation scenario based on SUMO.

FIGURE 3. The scenario of urban road with traffic light.

vehicle V_n can be calculated as

$$R_{n,m} = B\log(1+\gamma_{n,m}), \tag{15}$$

where B refers to the channel bandwidth, and we assume that the bandwidth is same for each vehicle V_n. If the vehicle V_n transmits its task with data size D_n to RSU_m, the data transmission delay can be calculated as

$$T^t_{n,m} = \frac{D_n}{R_{n,m}}. \tag{16}$$

Remark 2: The data transmission process must be completed during the time that the vehicle V_n is within the coverage of RSU_m, i.e., the dwell time $T^{t_{max}}_{n,m}$. That is to say, the transmission delay $T^t_{n,m}$ must be less then or equal to $T^{t_{max}}_{n,m}$. The representations of dwell time $T^{t_{max}}_{n,m}$ in different vehicular mobility models can be expressed as:

- Constant velocity model:

$$T^{t_{max}}_{n,m} = \frac{2r_m}{v_n}. \tag{17}$$

- Vehicle-following model:

$$T^{t_{max}}_{n,m} = \begin{cases} \dfrac{2r_m - d^v_n}{v_n} + t^v_n, & \text{if } V_n \text{ meets traffic light} \\ \dfrac{2r_m}{v_n}, & \text{otherwise} \end{cases}, \tag{18}$$

  here, $2r_m - d^v_n$ means the travelling distance in uniform motion phase, which is similar to (10).
- Travelling-time statistical model:

$$T^{t_{max}}_{n,m} = 2r_m\theta, \tag{19}$$

  here, we consider $d_n = 2r_m$ and the dwell time $T^{t_{max}}_{n,m}$ follows the Gamma distribution $Ga(2r_m, \theta)$.

Specially, if the task is offloaded to the MBS, then it will be processed by the remote cloud server. Thus, the transmission delay from the vehicle V_n to the MBS adds an additional delay T_{cloud}, which is composed of the transmission time from the MBS to the remote cloud server and the feedback time.

(a) The real-world map of Huangping Road in Beijing, China. (b) The simulation scenario based on SUMO.

FIGURE 4. The scenario of crooked road.

For the sake of simplification, denote T_{cloud} as an constant value. Then, the transmission delay from the vehicle V_n to the MBS is calculated as follows:

$$T^t_{n,0} = \frac{D_n}{R_{n,0}} + T_{cloud}. \tag{20}$$

where $R_{n,0}$ is the data transmission rate of the vehicle V_n which offload its task to the MBS.

C. THE TASK COMPUTATIONAL MODEL

When the task generated by V_n is offloaded to RSU_m, the task with computational request C_n will be processed by the co-located edge server or the remote cloud server. The computational delay can be expressed as

$$T^c_{n,m} = \frac{C_n}{\delta_m}, \tag{21}$$

where δ_m is the computational capability of RSU_m. Here, $m = 0$ means the task is processed by the remote cloud server.

D. HANDOVER MODEL

During the period of task processing, the vehicle V_n might move out of the coverage of RSU_m and enter into the coverage of another RSU $RSU_{m'}$, and $m' \geq m$. The computational result has to be transmitted firstly from RSU_m to $RSU_{m'}$, and then transmitted from $RSU_{m'}$ to the target vehicle V_n. Assuming that the data size of computational result is negligible compared to that of the task, and the feedback delay can be ignored. Then the handover delay is mainly related to the backhaul delay, which is expressed as

$$T^h_{m,m'} = (m' - m)c_t, \tag{22}$$

where c_t means the handover delay from RSU_m to RSU_{m+1}, which is assumed as a constant value. From (22), it can be seen that the critical issue is to predict where the vehicle V_n is located. From arriving at the coverage of RSU_m to the task has been finished, the travelling time of the vehicle V_n is $T_n = T^t_{n,m} + T^c_{n,m}$, which can be obtained from (16) and (21). Then the corresponding travelling distance d_n can be predicted by centralized controller under three different vehicular mobility models.

- Constant velocity model:

$$d_n = v_n T_n. \tag{23}$$

- Vehicle-following model:

$$d_n = \begin{cases} k'_n d^v_n + (T_n - k'_n t^v_n) v_n, & \text{if } V_n \text{ meets traffic light} \\ v_n T_n, & \text{otherwise} \end{cases}, \tag{24}$$

  here, $T_n - k'_n t^v_n$ means the moving time in uniform motion phase, which is easy to derive from (10).
- Travelling-time statistical model:

$$d_n = \frac{T_n}{\theta}, \tag{25}$$

  here, we treat T_n as the expectation value of the Gamma distribution $Ga(d_n, \theta)$, then (25) can be derived from (13).

When the vehicle travels from RSU_m to $RSU_{m'}$, the following inequality must be satisfied:

$$2\sum_{i=m}^{m'-1} r_i \leq d_n \leq 2\sum_{j=m}^{m'} r_j. \tag{26}$$

Then the location of the vehicle V_n can be predicted.

Remark 3: If the vehicle V_n is still within the coverage of RSU_m or it offloads its task to the MBS, there will be no handover cost, i.e, $T^h_{m,m} = 0$.

E. TOTAL OFFLOADING DELAY

Based on the discussion above, the total offloading delay can be expressed as the sum of transmission delay, the computational delay, the waiting delay and the handover delay, that

is

$$T_{n,m} = T^t_{n,m} + T^c_{n,m} + T^w_{n,m} + T^h_{m,m'}. \quad (27)$$

IV. PROBLEM FORMULATION

The optimization variable of the task offloading is defined as $x_{n,m}$. Here, $x_{n,m} = 1$ means the vehicle V_n offload its task to RSU_m, and otherwise, $x_{n,m} = 0$. Then the task offloading problem can be transformed into a matching problem between vehicles and RSUs. This work aims at minimizing the total offloading delay, which can be formulated as follows:

$$\begin{aligned}
\mathbf{P1}: \min_{\{x_{n,m}\}} & \sum_{m\in\mathcal{M},n\in\mathcal{N}} x_{n,m}T_{n,m}, \\
\text{s.t. } & C_1: x_{n,m}T_{n,m} \le \tau_n^{max}, \forall m \in \mathcal{M}, \forall n \in \mathcal{N}, \\
& C_2: T^t_{n,m} \le T^{t_{max}}_{n,m}, \forall m \in \mathcal{M}, \forall n \in \mathcal{N}, \\
& C_3: \gamma_{n,m} \ge \gamma_{min}, \forall m \in \mathcal{M}, \forall n \in \mathcal{N}, \\
& C_4: x_{n,m} \in \{0,1\}, \forall m \in \mathcal{M}, \forall n \in \mathcal{N}, \\
& C_5: \sum_{m\in\mathcal{M}} x_{n,m} \le 1, \forall n \in \mathcal{N}, \\
& C_6: \sum_{n\in\mathcal{N}} x_{n,m} \le q_m, \forall m \in \mathcal{M}. \quad (28)
\end{aligned}$$

Here, C_1 represents that the delay tolerate of the whole task offloading process. C_2 is the transmission delay constraint that the vehicle V_n must finish its transmission process during the dwell time in the coverage of RSU_m. C_3 denotes the QoS requirement in terms of SNR. $C_4 \sim C_6$ denote the task offloading relationship between vehicles and RSUs, which means that a vehicle can only offload its task to only one RSU, but the RSU can serve up to q_m vehicles, simultaneously.

V. TASK OFFLOADING BASED ON ONE-TO-ONE MATCHING

We firstly consider a simple scenario where each vehicle can offload its task to one RSU and each RSU merely execute the computational task of one vehicle, i.e., $q_m = 1$, $m \in \{1, 2, \cdots, M\}$. Then, the problem **P1** can be converted to a one-to-one matching [31] between N vehicles and M RSUs, which can be expressed as

$$\begin{aligned}
\mathbf{P2}: \min_{\{x_{n,m}\}} & \sum_{n\in\mathcal{N},m\in\mathcal{M}} x_{n,m}T_{n,m}, \\
\text{s.t. } & C_1 \sim C_5, \\
& C_7: \sum_{n\in\mathcal{N}} x_{n,m} \le 1, \forall m \in \{1, 2, \cdots, M\}, \quad (29)
\end{aligned}$$

where C_7 represents that any RSU can accept no more than one vehicle. The transformed problem is defined as a triplet $(\mathcal{V}, \mathcal{RSU}, \mathcal{F})$, where $\mathcal{V}$ and $\mathcal{RSU}$ are two finite and distinct sets of the participants in this matching, i.e., N vehicles and M RSUs, respectively. $\mathcal{F}$ denotes the set of the matching preference.

Definition 1 (One-to-One Matching): For the formulated matching problem $(\mathcal{V}, \mathcal{RSU}, \mathcal{F})$, a matching ϕ represents a one-to-one correspondence from the set $\mathcal{V} \cup \mathcal{RSU}$ onto the set $\mathcal{V} \cup \mathcal{RSU} \cup \{\emptyset\}$ based on the preference $\mathcal{F}$. $\phi(V_n) = RSU_m$ represents that the vehicle V_n is matched with RSU_m. Specially, if the vehicle V_n is rejected by all the RSUs, it will be matched with MBS, i.e., $\phi(V_n) = RSU_0$.

To carry out the matching, each vehicle is required to establish its preference list via arranging RSUs from the other side according to its preference. Denote $\mathcal{G} = \{G_1, ..., G_m, ...G_M\}$ as the price set of vehicles, in which G_m represents the matching cost of RSU_m. For any vehicle wishing to be matched with RSU_m, it has to bear the matching cost G_m. For the sake of simplicity, we define the preference of V_n towards RSU_m as the difference between the reciprocal of offloading delay $T_{n,m}$ and the matching cost G_m, which is given by

$$U_{n,m}\,|_{\phi(V_n)=RSU_m} = \frac{1}{T_{n,m}} - G_m. \quad (30)$$

It is noted that the initial value of $G_m \in \mathcal{G}$ is set as zero for simplification.

A complete, reflexive, and transitive binary preference relation, i.e., "$\succ$", is introduced to compare the preferences. For instance, the vehicle V_n prefers RSU_m to $RSU_{m'}$ can be represented as $RSU_m \succ_{V_n} RSU_{m'}$, $\forall n \in \mathcal{N}$, $m, m' \in \mathcal{M}$, and $m \ne m'$, which is given by

$$RSU_m \succ_{V_n} RSU_{m'} \Leftrightarrow U_{n,m} > U_{n,m'}. \quad (31)$$

Denote the preference list of V_n as $\mathcal{F}_n$, which is constructed by arranging all the M RSUs according to the obtained $U_{n,m}$ in a descending order. In the procedure of the one-to-one matching, N vehicles and M RSUs will be matched with each other in accordance with the derived preference lists.

The matching is implemented in an iterative manner. Any vehicle V_n that remains unmatched will send a matching request to its most preferred RSU_m in $\mathcal{F}_n$. If RSU_m receives only one request, then a matching between V_n and RSU_m will be constructed, i.e., $x_{n,m} = 1$. A matching conflict arises when RSU_m receives multiple matching requests from the vehicles simultaneously. In this case, the RSU_m will increase its price step by step. During the i-th pricing rising step, the price of RSU_m is given by

$$G_m[i] = G_m[i-1] + \Delta G, \quad (32)$$

where $G_m[i]$ means the price of RSU_m at the i-th pricing rising step. ΔG is a price increment which is a predefined amount. Afterwards, the preference lists of competing vehicles will be updated in accordance with the latest preference of RSU_m, which is decreased due to ΔG. As the matching cost increase, some competing vehicles will give up RSU_m if another more preferred RSUs occur, that is, these vehicles will prefer another RSU $RSU_{m'}$ than RSU_m ($m \ne m'$), i.e., $RSU_{m'} \succ_{V_n} RSU_m$. The price rising process will be finished when only one vehicle remains. The matching iteration will terminate when any vehicle $V_n \in \mathcal{V}$ has been matched with either an RSU or the MBS. The one-to-one matching is shown in Algorithm 1.

We give the definitions of the ***Blocking Pair*** and ***Stable Matching*** [42].

Algorithm 1 The Iterative Matching Algorithm

Input : $\mathcal{V}$, $\mathcal{RSU}$, ΔG ;
Output : ϕ;
Initialization :
Each vehicle $V_n \in \mathcal{V}$ builds its preference list $\mathcal{F}_n$ based on (31);
Set $\phi = \emptyset$, $P_m = 0$, $RSU_m^V = \emptyset$, $\forall m \in \mathcal{M}$,
while $\exists \phi(V_n) = \emptyset$ **do**
  for $V_n \in \mathcal{V}$ **do**
    Each vehicle $V_n \in \mathcal{V}$ proposes to the most preferred RSU RSU_m in its preference list $\mathcal{F}_n$;
  end for
  for $RSU_m \in \mathcal{RSU}$ **do**
    if RSU_m receives only one request **then**
      Match the vehicle V_n with RSU_m directly;
    end if
    if RSU_m receives more than one requests **then**
      Add V_n which proposed to RSU_m to the condition set RSU_m^V.
    end if
    if $RSU_m^V \neq \emptyset$ **then**
      while $RSU_m^V \neq \emptyset$ **do**
        Rising price G_m based on (32);
        The vehicle $V_n \in RSU_m^V$ update its corresponding preference list $\mathcal{F}_n$;
        if $V_n \in RSU_m^V$ has a better choice **then**
          Remove V_n from RSU_m^V.
        end if
      end while
    end if
  end for
end while

Definition 2 (Blocking Pair): The vehicle V_n and the RSU RSU_m form a blocking pair if both V_n and RSU_m prefer the others than their currently matched result.

Definition 3 (Stable Matching): A matching Φ is defined as stable if it is not blocked by any pair.

Theorem 1: Given the set of vehicles $\mathcal{V}$ and RSUs $\mathcal{RSU}$, Algorithm 1 achieves a stable matching between them.

Proof: Contradiction is utilized to verify the validity of *Proposition 1*. Assuming that the matching result is $\phi(V_n) = RSU_{m'}$, but V_n and RSU_m form a blocking pair, that is, V_n and RSU_m prefer to be matched each other, but they have not been matched, thus we have $\phi(V_n) \neq RSU_m$, $RSU_m \succ_{V_n} RSU_{m'}$. However, according to the pricing strategy of Algorithm 1, $\phi(V_n) = RSU_m$ is not the matching result, which means that the vehicle V_n has abandoned RSU_m during the process of rising prices. Furthermore, the final winner for V_n is $RSU_{m'}$, that is $RSU_{m'} \succ_{V_n} RSU_m$. The analysis result contradicts the assumption. Therefore, Algorithm 1 achieves a stable matching. ■

VI. TASK OFFLOADING BASED ON MATCHING WITH QUOTA

There is a practical scenario in which one RSU can accept multiple tasks generated by multiple vehicles. That is to say, each vehicle can offload its task to only one RSU but each RSU can execute the computational task from up to q_m vehicles, i.e., $q_m \geq 2$. Thus, the problem **P1** can be converted to a one-to-many matching [30], which is expressed as

$$\begin{aligned} \mathbf{P3}: \min_{\{x_{n,m}\}} & \sum_{m\in\mathcal{M}, n\in\mathcal{N}} x_{n,m} T_{n,m}, \\ \text{s.t. } & C_1 \sim C_5, \\ & C_8: \sum_{n\in\mathcal{N}} x_{n,m} \leq q_m, \forall m \in \{1, 2, \cdots, M\}, \end{aligned} \tag{33}$$

Here, C_8 represents that RSU_m can accept up to q_m tasks simultaneous. Similar to the one-to-one matching, the transformed problem **P3** is a one-to-many matching problem and can be defined as a triple $(\mathcal{V}, \mathcal{RSU}, \mathcal{F})$. $\phi(V_n) = RSU_m$ means that the V_n is matched with RSU_m. Specially, the vehicle V_n which is not matched with any RSU will be matched with the MBS.

In the one-to-many matching process, each vehicle proposes to its most preferred RSU based on the preference list $\mathcal{F}_n$, similar to the one-to-one matching process. If RSU_m receives no more than q_m computational request, it will accept all the proposed vehicles. Otherwise, RSU_m will increase its matching price G_m based on (32) until only q_m vehicles remain. The stability of one-to-many matching is different from that of one-to-one matching. In the one-to-many matching, we can use the concept of group stability. At first, a coalition $\mathcal{C} \subset \mathcal{V} \cup \mathcal{RSU}$ consists of at least one RSU. A matching ϕ is blocked by a coalition $\mathcal{C}$ if there exists another matching ϕ' that meets the following conditions:

- $\phi'(RSU_m) \in \mathcal{C}, \forall RSU_m \in \mathcal{C}$;
- $U_{n,m'} \mid_{\phi'(V_n)=RSU_{m'}} \geq U_{n,m} \mid_{\phi(V_n)=RSU_m}, \forall V_n \in \mathcal{C}$;
- If $V_n \in \phi'(\mathcal{RSU}_m)$, then $V_n \in \phi(RSU_m) \cup \mathcal{C}$.

The first condition ensures that all the vehicles V_n in $\mathcal{C}$ are matched to RSU_m in $\mathcal{C}$. The second conditions denotes that all vehicles in $\mathcal{C}$ prefer their current matching results in ϕ' to their matching results in ϕ, and the third condition represents that each vehicle can be matched with a combination of new RSU. Therefore, ϕ is blocked by some coalition $\mathcal{C}$, if the vehicle V_n and the RSU_m both find a better choice to ϕ. Given the above conditions, group stability is defined as follows.

Definition 4 (Group Stable Matching): A matching ϕ is defined as group stable if it is not blocked by any coalition.

Theorem 2: Given the set of vehicles $\mathcal{V}$, RSUs $\mathcal{RSU}$ and quota $q_m, \forall m \in \mathcal{M}$, the proposed algorithm achieves a group stable matching between $\mathcal{V}$ and $\mathcal{RSU}$.

Proof: Assuming that the matching result is ϕ, but it is blocked by a coalition $\mathcal{C}$, i.e., a matching ϕ' is better than current matching ϕ. Thus, it must have $U_{n,m'} \mid_{\phi'(V_n)=RSU_{m'}} \geq U_{n,m} \mid_{\phi(V_n)=RSU_m}, \forall V_n \in \mathcal{C}$. However, according to the

pricing strategy of the proposed algorithm, $\phi(V_n) = RSU_{m'}$ is not the matching result, which means that the vehicle V_n has abandoned $RSU_{m'}$ during the process of rising prices, and the winner for V_n is RSU_m. Thus, it must have $RSU_m \succ_{V_n} RSU_{m'}$. The analysis result contradicts the assumption. Therefore, the proposed matching algorithm achieves a stable matching result. ■

VII. NUMERICAL RESULTS

In this section, we evaluate the proposed algorithm based on different scenarios and vehicular mobility models. Firstly, we introduce the scenario establish and the experimental setting. Then, we present the numerical results. The simulation parameters can refer to previous work [43], and they are summarized in TABLE 2.

TABLE 2. Simulation parameters.

Variable	Value
Passloss exponent α	3.4
Transmission power P_t	30 dBm
Power of AWGN N_0	-114 dBm
Channel bandwidth B	10MHz
Data size of the task D_n	$150 \sim 250$ MB
Computational resource demand of vehicles C_n	$2.5 \sim 3$GB
Delay constraint τ_n	50s
Number of RSUs M	4
Number of vehicles N	$4 \sim 8$
Computational capability of RSU δ_m	$250 \sim 350$MHz
Velocity of vehicle v_n	$0 \sim 27.7$m/s
Acceleration and deceleration of vehicles a'_n, a_n	$0 \sim 4m/s^2$
Scope parameter of Gamma distribution θ	0.08
Unit handover delay c_t	2s

We adopt SUMO to evaluate the proposed algorithm based on real-world road topologies. The characteristic information of the real-world scenarios is extracted from OpenStreetMap, and using JOSM, which is an extensible editor for OpenStreetMap, to process the digital map. Then these digital map data are imported to SUMO for the processing, in which the vehicular traffics are generated based on the specific road topologies. Via the predefined interfaces of SUMO [35], some key parameters of vehicles, such as velocity and the corresponding time can be obtained during the simulation. These essential information will be saved as several XML files for data processing. The three simulation scenarios are presented as follows.

- **The Scenario of Straight Road:** It is a unidirectional straight lane which is the basic element in most traffic scenarios. A typical straight road, named Wugang Road, in Foshan City of the Guangdong Province, China, is selected for evaluation. Fig. 2(a) shows its real-world map based on aerial photography, and Fig. 2(b) shows its simulation scenario based on SUMO.
- **The Scenario of Urban Road with Traffic Light:** This scenario is based on the urban road with the traffic lights, and only considers the impact of the traffic lights on the vehicular velocity. A straight road with a traffic intersection in Jianguomen Overpass area in Beijing City, China, is selected. Fig. 3 (a)-(b) show its real-world map and the simulation scenario based on SUMO, respectively.
- **The Scenario of Crooked Road:** Complex road topology of roads, especially the road turning, is mainly considered in this scenario. We select a crooked road with a turning intersection in Changping District, Beijing City, China as the road simulation scenario. Fig. 4 (a)-(b) show its real-world map and the simulation scenario based on SUMO, respectively.

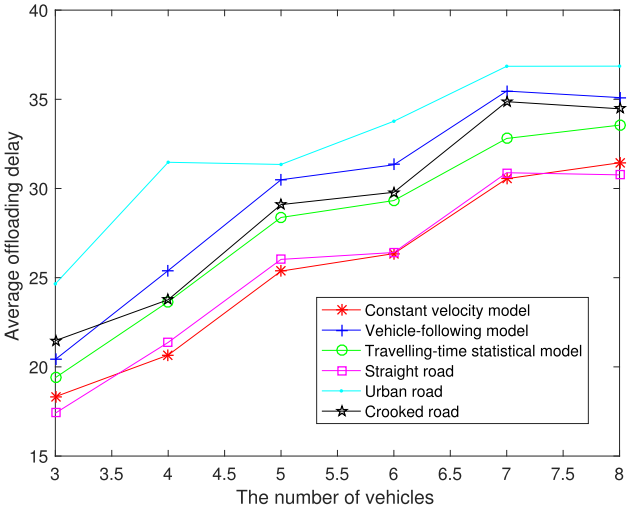


FIGURE 5. The average offloading delay versus the number of vehicles ($q_m = 1$).

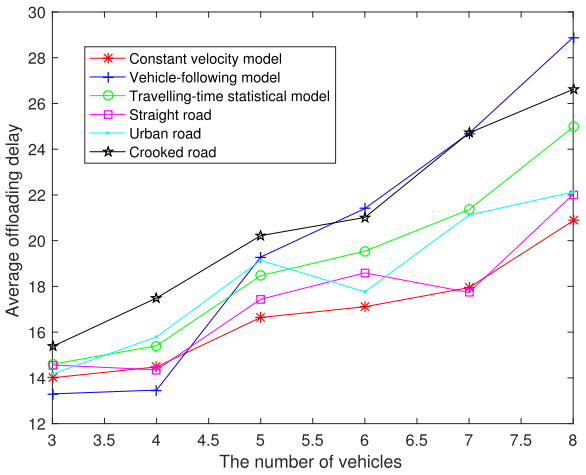


FIGURE 6. The average offloading delay versus the number of vehicles ($q_m = 2$).

Fig. 5 and Fig. 6 show the average offloading delay versus the number of vehicles under two different situations, which are formulated as one-to-one matching ($q_m = 1$) and one-to-many matching ($q_m = 2$), respectively. The number of RSUs is $M = 4$. It is clearly that the average delay increases monotonously with the number of vehicles N. The reason is that a larger number of vehicles leads to more tasks to be offloaded and processed, which increases both the

data transmission and task computation delay. Furthermore, the increasing number of vehicles may lead to a longer waiting delay due to competition in vehicles. It is obviously that the constant velocity model and the straight road scenario can achieve a smaller average delay compared with other models or scenarios. The reason is that they are more ideal and do not consider the influence of traffic light and turning. Thus, the average offloading delay is also smaller, since the waiting delay has a greater impact on average delay compared with transmission delay and handover delay. The traffic lights will have a large impact on the velocity of vehicles in vehicle-following model and the urban road scenario, which will cause a worse average offloading delay performance.

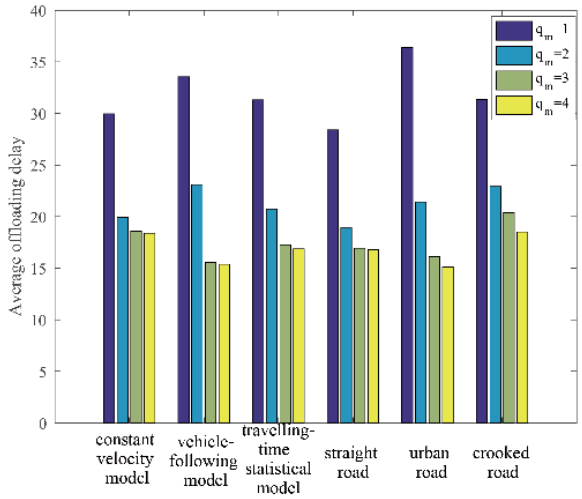


FIGURE 7. The average offloading delay in different models or scenarios ($N = 6$).

Fig. 7 shows the average offloading delay versus different models or scenarios under different quota values. The number of vehicles N is set to 6. The performances of $q_m = 1$ and $q_m = 2$ are consistent with the performances shown in Fig. 5 and Fig. 6, respectively. Specifically, we can note that the average offloading delay is inversely proportional to the quota value, i.e., a larger quota value gives rise to a smaller average delay. The reason is that the RSU can process more tasks when the quota value is large, which will reduce the waiting delay, and lead to a more efficient task offloading work.

Fig. 8 shows that the average waiting delay versus different models and scenarios under different quota values. It can be seen that with the increasing of quota, the waiting delay increases firstly and then decreases. Compared to Fig. 7, it can be seen although the waiting delay increases, the average offloading delay decreases. The reason is that when $q_m = 2$, more vehicles choose RSU rather than the MBS to offload their tasks, which will indeed reduce the offloading delay, but the waiting delay will increase due to the fact that the vehicle has to reach the target RSU based on matching results, which may be far away from the vehicle. With the quota further increasing, the vehicles choose nearby RSUs to offload theirs

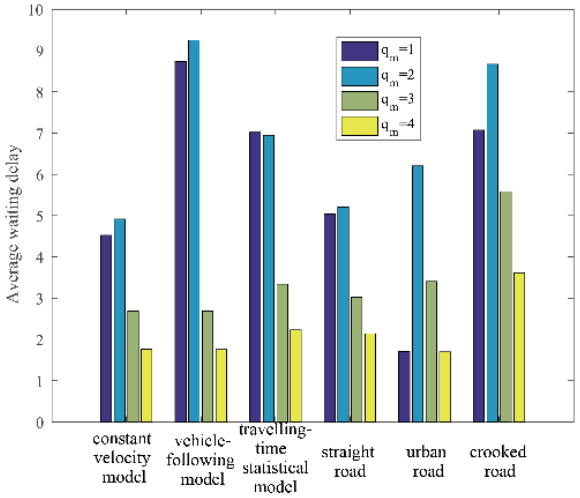


FIGURE 8. The average waiting delay in different models or scenarios ($N = 6$).

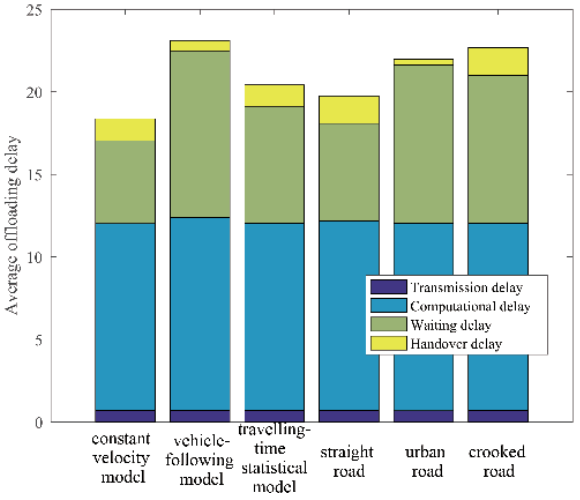


FIGURE 9. The average offloading delay in different models or scenarios ($N = 6$).

tasks, which reduces both the average offloading delay and the waiting delay.

Fig. 9 shows the proportion of transmission delay, computational delay, waiting delay and handover delay in the average offloading delay, with $N = 6$ and $q_m = 1$. It can be seen that the offloading delay is mainly dominated by waiting delay and computational delay. Given a task, and the computational delay is a determined value. Thus, how to reduce the task offloading delay mainly depends on how to reduce the waiting delay.

Fig. 10 and Fig. 11 show the average offloading delay versus the velocity of vehicle and task complexity, respectively. In Fig. 10, it can be seen that with the increasing of velocity, the average offloading delay is decreasing. It should be pointed out that the performance of travelling-time statistical model is little related to velocity, so it is a straight line. In Fig. 11, the average offloading delay increases almost

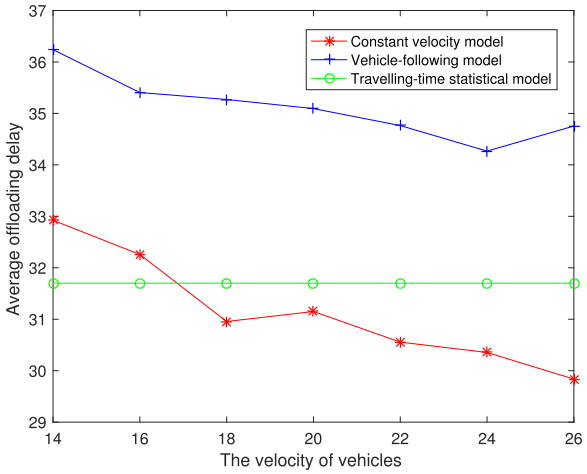


FIGURE 10. The average offloading delay versus the velocity of vehicle ($q_m = 1, N = 6$).

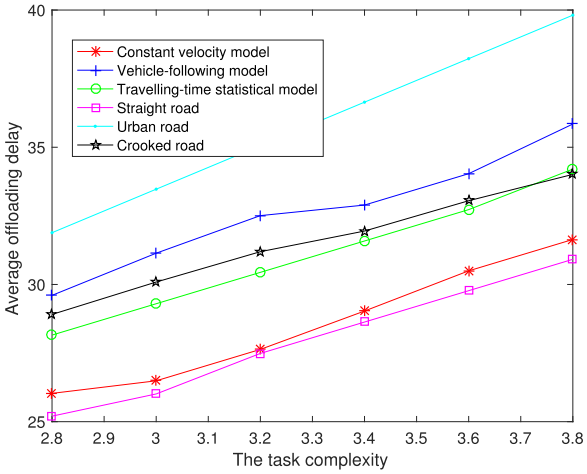


FIGURE 11. The average offloading delay versus the task complexity ($q_m = 1, N = 6$).

linearly with the task complexity. The reason is that the increasing delay is mainly related to the computational delay which is a linear function of the task complexity.

VIII. CONCLUSIONS

In this paper, we investigated the task offloading problem in vehicular edge computing, which aimed to minimize the task offloading delay. The task offloading delay was consist of data transmission delay, task computational delay, waiting delay, and handover delay, which was derived based on three different velocity models, i.e., constant velocity model, vehicle-following model, and travelling-time statistical model. Then a novel matching-based task offloading algorithm was proposed, and the original problem was transformed into one-to-one matching and matching with quota, respectively. The proposed algorithm was validated under three different simulation scenarios extracted by SUMO, which were straight road, urban road with traffic light, and crooked road. The numerical results showed that the proposed algorithm can effectively simulate the overall motion of the vehicle with suitable vehicular mobility models under different real-road topology scenarios, and achieve a significant delay decreasing.

REFERENCES

[1] K. M. Alam, M. Saini, and A. E. Saddik, "Toward social Internet of vehicles: Concept, architecture, and applications," *IEEE Access*, vol. 3, pp. 343–357, Mar. 2015.
[2] M. A. Salahuddin, A. Al-Fuqaha, and M. Guizani, "Software-defined networking for RSU clouds in support of the Internet of vehicles," *IEEE Internet Things J.*, vol. 2, no. 2, pp. 133–144, Apr. 2015.
[3] N. Sathishkumar and K. Rajakumar, "A study on vehicle to vehicle collision prevention using fog, cloud, big data and elliptic curve security based on threshold energy efficient protocol in wireless sensor network," in *Proc. ICRTCCM*, Kansas, MO, USA, Feb. 2017, pp. 275–280.
[4] Y. Leng and L. Zhao, "Novel design of intelligent Internet-of-vehicles management system based on cloud-computing and Internet-of-Things," in *Proc. ICC*, Harbin, China, Aug. 2011, pp. 3190–3193.
[5] Z. Zhou, M. Dong, K. Ota, G. Wang, and L. T. Yang, "Energy-efficient resource allocation for D2D communications underlaying cloud-RAN-based LTE-A networks," *IEEE Internet Things J.*, vol. 3, no. 3, pp. 428–438, Nov. 2015.
[6] H.-L. Truong and M. Karan, "Analytics of performance and data quality for mobile edge cloud applications," in *Proc. IEEE 7th Int. Conf. CLOUD*, San Francisco, CA, USA, Jul. 2018, pp. 2159–6190.
[7] C.-M. Huang, M.-S. Chiang, D.-T. Dao, W.-L. Su, X. Xu, and H. Zhou, "V2V data offloading for cellular network based on the software defined network (SDN) inside mobile edge computing (MEC) architecture," *IEEE Access*, vol. 6, pp. 17741–17755, Mar. 2018.
[8] Z. Zhou, C. Gao, C. Xu, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Social big-data-based content dissemination in internet of vehicles," *IEEE Trans Ind. Informat.*, vol. 14, no. 2, pp. 768–777, Jul. 2017.
[9] P. Zhao, H. Tian, C. Qin, and G. Nie, "Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing," *IEEE Access*, vol. 5, pp. 11255–11268, Jun. 2017.
[10] C. Gong, M. Li, L. Zhao, Z. Guo, and G. Han, "Homomorphic evaluation of the integer arithmetic operations for mobile edge computing," *Wireless Commun. Mobile Comput.*, vol. 2018, Nov. 2018, Art. no. 8142102.
[11] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, Mar. 2017.
[12] E. Uhlemann, "Initial steps toward a cellular vehicle-to-everything standard [connected vehicles]," *IEEE Trans. Veh. Technol.*, vol. 12, no. 1, pp. 14–19, Feb. 2017.
[13] L. Du and H. Dao, "Information dissemination delay in vehicle-to-vehicle communication networks in a traffic stream," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 66–80, Feb. 2015.
[14] J. Liu and Q. Zhang, "Offloading schemes in mobile edge computing for ultra-reliable low latency communications," *IEEE Access*, vol. 6, pp. 2169–3536, Feb. 2018.
[15] L. Liu and W. Yu, "A D2D-based protocol for ultra-reliable wireless communications for industrial automation," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5045–5058, Aug. 2018.
[16] Z. Zhou, J. Feng, C. Zhang, Z. Chang, Y. Zhang, and K. M. S. Huq, "SAGECELL: Software-defined space-air-ground integrated moving cells," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 92–99, Aug. 2018.
[17] N. Abbas, Y. Zhang, A. Taherkord, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Sep. 2017.
[18] A. Rahman, E. Hassanain, and M. S. Hossain, "Towards a secure mobile edge computing framework for Hajj," *IEEE Access*, vol. 5, pp. 11768–11781, Jun. 2017.
[19] A. C. Baktir, A. Ozgovde, and C. Ersoy, "How can edge computing benefit from software-defined networking: A survey, use cases, and future directions," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2359–2391, Jun. 2017.
[20] Z. Zhou, H. Liao, B. Gu, K. Huq, S. Mumtaz, and J. Rodriguez, "Robust mobile crowd sensing: When deep learning meets edge computing," *IEEE Netw.*, vol. 32, no. 4, pp. 54–60, Aug. 2018.
[21] Y. Liu, S. Wang, J. Huang, and F. Yang, "A computation offloading algorithm based on game theory for vehicular edge networks," in *Proc. IEEE ICC*, Kansas, MO, USA, Jul. 2018, pp. 1–6.

[22] M. Ali, N. Riaz, M. Ashraf, S. Qaisar, and M. Naeem, "Joint cloudlet selection and latency minimization in fog networks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4055–4063, Apr. 2018.
[23] B. Gu, Y. Chen, H. Liao, Z. Zhou, and D. Zhang, "A distributed and context-aware task assignment mechanism for collaborative mobile edge computing," *Sensors*, vol. 18, no. 8, p. 2423, Aug. 2018.
[24] G. Qiao, S. Leng, K. Zhang, and Y. He, "Collaborative task offloading in vehicular edge multi-access networks," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 48–54, Aug. 2018.
[25] Z. Zhou, P. Liu, Z. Chang, C. Xu, and Y. Zhang, "Energy-efficient workload offloading and power control in vehicular edge computing," in *Proc. IEEE WCNCW*, Barcelona, Spain, Jul. 2018, pp. 191–196.
[26] Y. Gu, W. Saad, M. Bennis, M. Debbah, and Z. Han, "Matching theory for future wireless networks: Fundamentals and applications," *IEEE Commun. Mag.*, vol. 53, no. 5, pp. 52–59, May 2015.
[27] Z. Zhou *et al.*, "When mobile crowd sensing meets UAV: Energy-efficient task assignment and route planning," *IEEE Trans. Commun.*, vol. 66, no. 11, pp. 5526–5538, Jul. 2018.
[28] Y. Al-Dubai, L. Zhao, A. Y. Zomaya, and G. Min, "QoS-aware inter-domain multicast for scalable wireless community networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 11, pp. 3136–3148, Nov. 2014.
[29] L. Zhao, A. Al-Dubai, X. Li, G. Chen, and G. Min, "A new efficient cross-layer relay node selection model for wireless community mesh networks," *Comput. Elect. Eng.*, vol. 61, pp. 361–372, Jul. 2017.
[30] S. Bayat, Y. Li, L. Song, and Z. Han, "Matching theory: Applications in wireless communications," *IEEE Signal Process. Mag.*, vol. 33, no. 6, pp. 103–122, Nov. 2016.
[31] L. Xu, C. Jiang, Y. Shen, T. Q. S. Quek, Z. Han, and Y. Ren, "Energy efficient D2D communications: A perspective of mechanism design," *IEEE Trans. Wireless Commun.*, vol. 15, no. 11, pp. 7272–7285, Nov. 2016.
[32] B. Wang, Y. Sun, S. Li, Q. Cao, Y. Chen, and J. Xu, "Hierarchical matching with peer effect for latency-aware caching in social IoT," in *Proc. IEEE SmartIoT*, Xi'an, China, Aug. 2018, pp. 255–262.
[33] J. Zhao, Y. Liu, K. Chai, Y. Chen, and M. Elkashlan, "Many-to-many matching with externalities for device-to-device communications," *IEEE Wireless Commun. Lett.*, vol. 6, no. 1, pp. 138–141, Dec. 2017.
[34] Y. Gu, Y. Zhang, M. Pan, and Z. Han, "Matching and cheating in device to device communications underlying cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 10, pp. 2156–2166, Oct. 2015.
[35] L. Bedogni, M. Gramaglia, A. Vesco, M. Fiore, J. Härri, and F. Ferrero, "The bologna ringway dataset: Improving road network conversion in SUMO and validating urban mobility via navigation services," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5464–5476, Dec. 2015.
[36] L. Codeca, R. Frank, S. Faye, and T. Engel, "Luxembourg SUMO traffic (LuST) scenario: Traffic demand evaluation," *IEEE Trans. Intell. Transport. Syst.*, vol. 9, no. 2, pp. 52–63, Apr. 2017.
[37] Z. Zhou, H. Yu, C. Xu, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Dependable content distribution in D2D-based cooperative vehicular networks: A big data-integrated coalition game approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 953–964, Mar. 2018.
[38] J. Song, Y. Wu, Z. Xu, and X. Lin, "Research on car-following model based on SUMO," in *Proc. IEEE 7th Int. Conf. Adv. Infocomm Technol. (ICAIT)*, Nov. 2014, pp. 47–55.
[39] R. Jiang, Y. Zhu, T. He, Y. Liu, and L. M. Ni, "Exploiting trajectory-based coverage for geocast in vehicular networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 12, pp. 3177–3189, Dec. 2014.
[40] R. Russell and T. Urban, "Vehicle routing with soft time windows and Erlang travel times," *J. Oper. Res. Soc.*, vol. 59, no. 9, pp. 1220–1228, 2007.
[41] I. Kaparias, M. G. H. Bell, and H. Belzner, "A new measure of travel time reliability for in-vehicle navigation systems," *J. Intell. Transp. Syst.*, vol. 12, pp. 202–211, Nov. 2008.
[42] Z. Zhou, K. Ota, M. Dong, and C. Xu, "Energy-efficient matching for resource allocation in D2D enabled cellular networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 5256–5268, Jun. 2016.
[43] Z. Zhou, H. Yu, C. Xu, Z. Chang, S. Mumataz, and J. Rodriguez, "BEGIN: Big data enabled energy-efficient vehicular edge computing," *IEEE Commun. Mag.*, vol. 56, no. 126, pp. 82–89, Dec. 2018.

PENGJU LIU is currently pursuing the bachelor's degree with North China Electric Power University, China. His research interests include resource allocation and energy management in VEC.

JUNLUO LI is currently pursuing the bachelor's degree with North China Electric Power University, China. Her research interests include the energy Internet systems and allocation problem.

ZHONGWEI SUN received the Ph.D. degree from Northwestern Polytechnical University, Xi'an, China, in 1999. He is currently an Associate Professor with the School of Electrical and Electronic Engineering, North China Electric Power University, Beijing, China. His research interests include smart grid and intelligent transportation systems, with the emphasis on information security.