JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 36, 865-878 (2020)
DOI: 10.6688/JISE.202007_36(4).0011

RAVEC: An Optimal Resource Allocation Mechanism in Vehicular MEC Systems

GAO-FENG HONG, WEI SU, QI-LI WEN AND PENG-LEI WU
School of Electronic and Information Engineering
Beijing Jiaotong University
Beijing, 100044 P.R. China
E-mail: {honggf; wsu; 17120137; 17120141}@bjtu.edu.cn

The development of vehicular services in Internet of vehicles poses challenges for vehicles with limited computation resources to guarantee the quality of service (QoS) of latency-sensitive and massive computation onboard services. Vehicular mobile edge computing (VEC) has emerged as an effective technology to enhance vehicular service quality through offloading onboard computation tasks to mobile edge computing (MEC) servers. MEC technology can reduce task processing latency and data transmission latency through its on-premises feature. However, the deployment of VEC still faces several problems such as lacking rational and effective resource allocation schemes. In order to solve these problems, we provide an optimal resource allocation mechanism in vehicular MEC systems (RAVEC) to minimize the total task processing delay among a set of vehicles in a time slot by using a global optimization perspective. The method considers the computation ability of each MEC server at road side unit (RSU) in a road segment, the mobility of each vehicle and the total offloading latency of a set of vehicles to get a best resource allocation plan and achieve onboard task offloading. Simulation results show that RAVEC demonstrates a reliable solution and has a certain value for future research.

Keywords: vehicular network, mobile edge computing, resource allocation, task offloading, global optimization

1. INTRODUCTION

The internet of vehicles (IOV) is emerging thanks to the advance of computing and communication technology. More and more researchers have been focusing on vehicular ad hoc network (VANET) during the past few years [1, 2]. Vehicle to infrastructure (V2I), vehicle to vehicle (V2V), vehicle to the Internet (V2N) and vehicle to pedestrian (V2P) are the typical communication paradigms in VANET and the researchers want to implement V2X which means vehicles can communicate with everything. IEEE 802.11p VANET and 5G-based cellular networks provide supports for IOV on aspects of communication [3]. Vehicular task computing is a crucial element to achieve new technologies in the field of intelligent traffic including traffic prediction, autonomous driving, collision avoidance, bird's eye view, automated overtake *etc*. [4]. Because emerging traffic services often have low latency requirements and require a large amount computation resources [5], the current onboard computation capacity may not meet these requirements. Considering the communication aspects, the feature of conventional cloud computing may bring considerable overhead to the backbone network when vehicular users (VUs) offload

Received January 3, 2020; revised February 4, 2020; accepted February 17, 2020.
Communicated by Xiaohong Jiang.

computation task to the servers [6]. In other words, there is always a long distance for data transmission between VUs and remote cloud servers, and it is difficult to guarantee the quality of service (QoS) and quality of experience (QoE) requirements [7].

Mobile edge computing (MEC) technology is introduced into IOV to solve the above problems. MEC extends onboard task computing power to the vicinity of VUs [8], with the help of MEC-enabled task offloading technology, VUs can obtain higher task processing speed and lower response latency. However, the MEC servers usually face problems of limited resource. Moreover, the communication overhead(bandwidth *etc*.) in the data offloading [9] also exists in computation offloading [10]. So it's important to propose an efficient resource allocation scheme in MEC-based vehicular networks to improve the experience of VU.

According to previous researches, the task offloading process in VEC generally has three parts: (1) Task offloading: The data packet with computation tasks is sent to the MEC servers by vehicular onboard device via V2I communication paradigm. There are two main factors which may affect the transmission latency: the size of the data packet and the condition of the wireless channel; (2) Task calculation: After the MEC servers at RSUs receive the task from vehicles, they will decide the destination processing server of the task according to the relevant resource allocation scheme. The task will be sent to the destination server and the calculation will be completed; (3) Feedback of task computation results: After the completion of the task calculation, the result will be sent back to the onboard device. Although the relevant schemes of task offloading have been analyzed and optimized for many years in academia, the existing schemes are still flawed. There is a lack of an efficient scheme to allocate the limited computation resources so that the total system performances can not be enhanced. Based on the existing researches [11–13], we mainly consider how many computation resources each MEC server can share. The computing capacity of each MEC server is dynamic changing which may influence the task processing performance. We also find that the location and mobility of vehicle may change the data transmission rate of V2I communication. Thus, when choosing the destination server to offload the onboard of a vehicle, its location, speed, the latency threshold of the task and the states of each MEC server should all be taken into consideration. In summary, the main contributions of this paper are as follows:

- Considering the computing capacity of each MEC server may change in different time slots, we firstly divide MEC servers at RSUs into several types according to their current computation resource sharing ability. This step can can motive MEC servers with large computing ability and low resource occupancy rate to share their computation resources for vehicular onboard task offloading.

- An optimal offloading scheme in the multiple MEC servers scenario with the consideration of the location and mobility of each vehicle is proposed. The task offloading decisions are jointly optimized.

- In order to minimize the total task offloading delay of a set of vehicles in a time slot, a comprehensive optimization algorithm is proposed to derive an optimal resource distributed method.

The rest of this paper is organized as follows. We will review the related work in Section 2. Then in Section 3, we will describe the design of RAVEC. Simulation and evaluation of the mechanism will be given in Section 4. And we will summarize our paper in Section 5.

2. RELATED WORK

MEC is attracting a lot attentions of academia and industry. It has great advantages to cope computation tasks with low-latency and high computation resource demand because of its on-premises feature. X. Lyu *et al*. consider the transmit power and CPU frequency of each device to optimize the offloading decisions as well as propose a model with weighting methods to achieve energy consumption and task offloading latency minimization [14]. In [15], the authors propose a one-to-many framework(one mobile device and multiple edge devices) and optimize the offloading decisions by considering the CPU frequency of APs. The authors also verify that task offloading decisions can be affected by the channel status. For a better utilization of local and MEC computation resources, a partial offloading model is proposed by Ren J *et al*. in [16], this model divides onboard tasks into two parts, one part is offloaded to the MEC server while the other part can be processed onboard. In [17] a hybrid MEC offloading scheme is proposed to minimize the multiuser video compression offloading latency. Due to the deployment and maintenance cost, the computational resources of MEC servers are limited. The MEC servers may relay the computation tasks to the remote cloud [18, 19], or other around servers [20, 21] and reduce the workloads to ensure the QOS when the task computation is huge. They also considered the tradeoffs between the transmission latency and task processing latency. However, the devices are not in high moving speeds in the previous researches, the previous schemes in MEC-enabled networks may not be applied in VEC due to the mobility feature of vehicles.

Recently, several conventional offloading schemes have been optimized and extended to the VEC networks [22-26]. The vehicles and roadside BSs are divided into a two-level architecture [22]. K. Wang *et al*. investigated the collaborative MEC framework in IOV to solve the limited computational capacity problems in MEC [23]. In [24], a game theory is used in the offloading scheme to minimize the latency. Y. Liu *et al*. proposed an autonomous VEC framework, they use GPS information to team up the vehicles and design a task caching scheme to optimize the task offloading [25]. In [26], Zhang *et al*. considered task latency tolerance and the computation resource limitation in vehicular networks and proposed a contractbased resource allocation mechanism to maximize the profit of the MEC server provider. However, all of the above schemes lack a uniform framework to solve the limited computation resource allocation problem in VEC from the perspectives of minimizing the overall network delay. Specifically, an appropriate method to offload the onboard task to the most suitable MEC server according to the current computation ability of each server requires further investigation. Also, the wireless transmission distance will mainly influence the data transmission rate in V2I communication paradigm. Thus, when proposing an optimal scheme to minimize the total system costs, all of these related factors should be taken into consideration.

3. DESIGN OF RAVEC

In this section, we will show how RAVEC works by presenting the VEC framework including MEC classifying model, the communication process, optimal offloading algorithm and formulations of the model.

3.1 VEC Framework

We consider that RSUs are deployed evenly along a straight road. Each RSU has a collocated MEC server, and the total available computation resource of these servers can

be different for vehicles may occupy part of the resource to offload their computation task. A Data center is deployed in the upper layer network and connected with MEC servers to get information from all these servers. The framework is shown in Fig. 1.

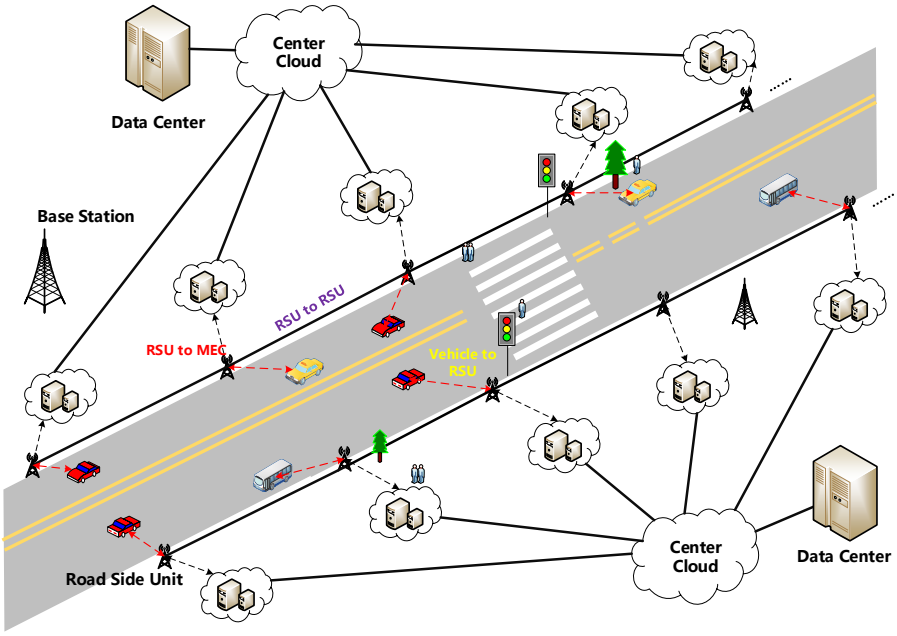


Fig. 1. MEC-enabled vehicular network senario.

In each road section, RSUs take charge of vehicular communication resource information exchange and task assignment. If the onboard unit can't satisfy the computation demands of the vehicle, RSUs equipped with MEC servers act as computational nodes and assist vehicles in task computation.

Due to the mobility of vehicles, the location of vehicles may change every once in a while and will affect the data transmission rate during the task offloading process. When task arrives, the MEC servers will decide whether send it to other servers on the road section judging from their current computation ability and the demand of computation resource for the task. Also, when RSUs send task processing results back to vehicles, the distances between the RSUs and vehicles should be considered.

In V2I communication paradigm, the wireless network states will be periodically broadcasted by RSUs. Vehicles will choose the spectrum band according to the information from RSUs. They transmit data in different spectrum and each vehicle has an orthogonal spectrum channel to avoid the co-channel interference.

3.2 MEC Classifying Model

To simplify our model, we divide continuous time into discrete intervals [27]. Our offloading process is calculated in a slot-by-slot fashion. Vehicles drive into the road section is donated as set $V_M\{v_1, v_2, \cdots, v_m, \cdots, v_M\}$ and the set of RSUs on a specific road section remains fixed within each slot which is denoted as $R_N\{r_1, r_2, \cdots, r_n, \cdots, r_N\}$. We use a set $\{D_m, C_m, \gamma_m\}$ to describe the onboard task feature of vehicle v_m, where D_m is the data size of the task, C_m is the computation resource which the task needed, and γ_m is the

task processing latency limitation. We define f_{v_m} as the onboard computation capacity of vehicle v_m and the velocity of vehicle v_m as S_{v_m}.

Since the number of MEC servers on the road section is fixed, the number of MEC server types is in a finite space.

We denote the set of MEC servers on the given road section as $S_K\{s_1, s_2, \cdots, s_k, \cdots, s_K\}$ and present the resource sharing capability of each type in a set $\Theta_K\{\theta_1, \theta_2, \cdots, \theta_k, \cdots, \theta_K\}$, where

$$\theta_1 < \theta_2 < \cdots < \theta_k < \cdots < \theta_K, \quad k \in K \tag{1}$$

Here, we will describe how to get the type of each MEC server and express it in formula. The computation size of the computation task to be processed on MEC server at the RSU r_n is C_n . If the computation resource is shared with a vehicle, the task processing delay on the sever will be increased, which is shown as

$$\Delta\gamma_n = \frac{C_n\alpha_n}{\alpha_{n,0}^2 - \alpha_n\alpha_{n,0}} + \varepsilon_{n,t} \leq \Delta\gamma_{n,max}, \tag{2}$$

where $\alpha_{n,0}$ is the rest available computation resource, α_n is the occupied computation resource, $\varepsilon_{n,t}$ is the delay caused by the external environment affecting the server at time slot t. Ideally, the value of $\varepsilon_{n,t}$ would be 0. The formula shows that the increased delay caused by the resource sharing shouldn't be more than a threshold $\Delta\gamma_{n,max}$ to guarantee the QoE and QoS requirements.

By deriving formula 3, we can find the upper bound of α_n as

$$\alpha_n^{upper} = \frac{\alpha_{n,0}^2(\Delta\gamma_{n,max} - \varepsilon_{n,t})}{\alpha_{n,0}(\Delta\gamma_{n,max} - \varepsilon_{n,t}) + C_n}, \tag{3}$$

where α_n^{upper} is the upper bound of computation resources that can be shared whose value is in a continuous closed interval $[\alpha_{min}, \alpha_{max}]$. We divide the interval into K equal-length subintervals. θ_k is the lower bound of the subinterval k, which can be calculated according to the equality below

$$\theta_k = \alpha_{min} + \frac{k-1}{K}(\alpha_{max} - \alpha_{min}). \tag{4}$$

The type of server on the RSU r_n is noted as s_k if $\theta_k \leq \alpha_n^{upper} < \theta_{k+1}$. When the task arrives, it will decide the relevant MEC server on the road section to process the task based on the MEC type and the computation demand of the task.

3.3 Communication Process

In our scenario, each RSU in the set R_N has a state table of the type of MEC server attached to the RSU in the set, the information of each RSU is defined as a tuple $\{r_n, \theta_k\}$ in the table and RSUs in the set periodically communicate with each other r_n to update the type information. When a vehicle enters the section of the road, it will judge whether to compute the task onboard locally according to the inequality below

$$\frac{D_m}{f_{v_m}} < \gamma_m, \tag{5}$$

if the task onboard satisfies the inequality, the task will be completely computed locally. Otherwise, the vehicle will send a request to the nearest RSU to get the state table of the type of MEC servers on the road section.

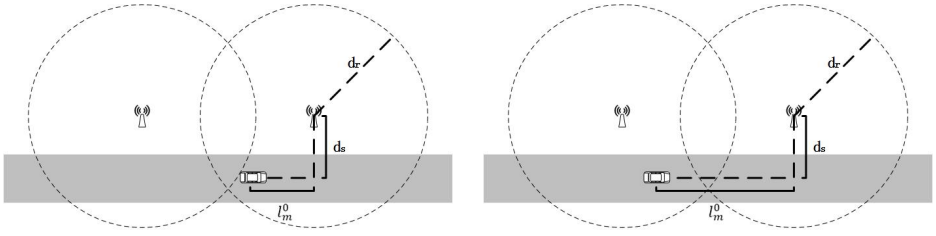
As the vehicle moves closer to the RSU, the data transmission rate becomes higher and the transmission latency becomes lower. In order to minimize the transmission latency, the vehicle should choose a optimal transmission time t_m. At the transmission time t_m, the data transmission rate follows

$$R_m = B_{c,m} \log_2 (1 + \frac{P_m d_{m,c}^{-\delta} h_{m,c}^2}{N_0}) \tag{6}$$

where $B_{c,m}$ is the link bandwidth between vehicle v_m and RSU r_c, rayleigh channel coefficient is presented as $h_{m,c}$, N_0 is the power noise, the path loss exponent is noted as δ, $d_{m,c}$ is the distance between the vehicle v_m and RSU r_c which satisfies

$$\begin{cases} \sqrt{(l_m^0)^2 + d_s^2}, & 0 < l_m^0 \leq d_r \\ \sqrt{(l_m^0 - S_{v_m} t_m)^2 + d_s^2}, & l_m^0 > d_r \end{cases} \tag{7}$$

where d_r is the coverage radius of the RSU, d_s is the vertical distance between RSU and the center of the road, l_m^0 is the distance between the vehicle v_m and the vertical mapping point from the first RSU of the driving direction to the center of the road (When a new time slot starts, if the vehicle has passed the vertical mapping point of the current RSU, it will choose the next RSU in its driving direction as the first entering RSU), the two cases in formula 7 are shown in Figs. 2 (a) and (b).



(a) $0 < l_m^0 \leq d_r$ (b) $l_m^0 > d_r$

Fig. 2. The location of vehicle v_m.

3.4 Optimal Offloading Algorithm

In each time slot, we divide vehicles in the vehicle set into two group. When $\frac{D_m}{f_{v_m}} < \gamma_m$, vehicle v_m belongs to group G_L. Otherwise, vehicle v_m belongs to group G_O. To minimize the total offloading latency of vehicles in group G_O, we analyze the whole offloading process of each vehicle and give a optimal resource allocation algorithm.

Due to the mobility feature of vehicles, the vehicles may leave out of the coverage area of the RSU, causing the data transmission distances between the vehicles and RSU to increase. In this article we set the transmission time t_m as the moment vehicle v_m enters the coverage of the first RSU of its driving direction to let the vehicle send its data out as soon as possible. After servers receive the task from vehicles, they will calculate the total offloading delay as the following step:

A. Task transmission delay

The task of vehicle v_m in group G_O should be offloaded to the MEC server. First, the task should be transmitted from the vehicle to the closest RSU r_c. The task transmission delay between the vehicle and the nearest RSU can be written as

$$T_{c,m}^{m,t} = \frac{D_m}{R_m}. \tag{8}$$

After the nearest RSU r_c receives the task, it will transmit the task to the destination RSU r_n. The channel bandwidth between adjacent RSUs is fixed, which is defined as B [28]. Hence the transmission time of the task between the nearest RSU r_c to the destination RSU r_n is

$$T_{c,n}^{m,t} = \frac{D_m}{B}(n-c), \tag{9}$$

where $n-c$ is the hops between r_c and r_n. The size of the data transmission from the MEC server to vehicle v_m is very small, so the transmission time can be ignored [29]. The total task transmission delay is

$$T_{n,m}^{T} = T_{c,m}^{m,t} + T_{c,n}^{m,t}. \tag{10}$$

B. Task processing delay

The resource sharing capability of RSU r_n is θ_k and C_m is the computation resource the vehicle v_m needed according to the previous description. The task processing delay when vehicle v_m offloads its task to RSU r_n is calculated as

$$T_{n,m}^{p} = \frac{C_m}{\theta_k}. \tag{11}$$

Here, we build a $N*M$ matrix X to represent the whole offloading task decision in the time slot t between N RSUs and vehicles in group G_O. Each element $x_{n,m}$ in matrix X can only equal to 0 or 1. $x_{n,m} = 1$ represents the onboard task of vehicle $v_m(m \in G_O)$ is processed on RSU r_n and $x_{n,m} = 0$ otherwise. Considering that the offloading process of each vehicle will produce certain computation loads on the target MEC server based on the previous server typing algorithm results. The task processing time of each server will be less than that without considering processing performance loss. So we define the current task processing ability weight of the target MEC server on RSU r_n as ω_n. The value of ω_n becomes larger as the shared resource of the server on RSU r_n increases. The optimization problem is formulated as P1.

$$\mathbf{P1}: \min_{X} \sum_{n \in R_N} \sum_{m \in G_O} (1 + 0.2 * tanh(\omega_n)) x_{n,m} (T_{n,m}^{T} + T_{n,m}^{p}),$$

s.t

$$C_1: \sum_{r_n \in R_N} x_{n,m} \leq 1, \forall r_n \in R_N$$

$$C_2: \frac{\sqrt{d_{m,c}^2 - d_s^2} + d_r(n-c)}{S_{v_m}} > T_{n,m}^{T} + T_{n,m}^{p}$$

$$C_3: T_{n,m}^{T} + T_{n,m}^{p} < \gamma_m$$

$$C_4: c \leq n$$

where C_1 represents there is a one-to-many correspondence among RSUs and vehicles. Considering the mobility of each vehicle, task processing on RSU r_n may not finish when vehicle v_m moves out of its communication range , which causes the offloading failure. The offloading scheme should meet the time delay limitation C_2. The task processing latency limitation should also be taken into consideration, so the total offloading latency should also satisfy the inequality C_3. C_4 represents vehicles only consider RSUs ahead of them to offload their tasks.

We present the whole multiple conditional cycles process in the Algorithm 1. RSUs are in charge of the computation resource assignment. Each vehicle uploads its state parameter to its nearest RSU, and then the whole offloading task decision between vehicles and MEC servers is calculated by the RSUs based on the algorithm. Eventually, RSUs on the section of the road broadcast the matching result to the set of vehicles, so that vehicles can offload their tasks to the corresponding MEC servers accordingly.

Algorithm 1

Require:

The parameters of each elements in set V_M and set R_N;

Stage 1: MEC Classifying

if $\theta_k \leq \frac{\alpha_{n,0}^2(\Delta\gamma_{n,max}-\varepsilon_{n,t})}{\alpha_{n,0}(\Delta\gamma_{n,max}-\varepsilon_{n,t})+C_n} < \theta_{k+1}$ **then**

  The type of the MEC server is s_k;

end if

Stage 2: Calculate whether offloading its task to MEC servers based on $\frac{D_m}{f_{vm}} < \gamma_m$

for $v_m \in V_M$ **do**

  if $\frac{D_m}{f_{vm}} < \gamma_m$ **then**

    put v_m into group G_L ;

    compute the task onboard locally ;

  else

    put v_m into group G_O ;

  end if

end for

Stage 3: Resource allocation

for $v_m \in G_O$ **do**

  for $r_n \in R_N$ **do**

    if $\frac{\sqrt{d_{m,c}^2-d_s^2}+d_r(n-c)}{S_{vm}} > T_{n,m}^T + T_{n,m}^p$ and $T_{n,m}^T + T_{n,m}^p < \gamma_m$ **then**

      Calculate the task transmission delay $T_{n,m}^T$ based on (8)(9)(10);

      Calculate the task processing delay $T_{n,m}^p$ based on (11);

      Save the value $T_{n,m}^T + T_{n,m}^p$;

    end if

  end for

end for

Calculate $\min_X \sum_{n\in R_N}\sum_{m\in G_O}(1+0.2*tanh(\omega_n))x_{n,m}(T_{n,m}^T+T_{n,m}^p)$ according to the values above;

Ensure:

The optimal resource allocation result;

3.5 Computational Complexity

Our MEC servers classifying scheme has N equality constraints and $2N+1$ inequality constraints and can be viewed as a convex programming problem. The computational complexity of this process is $\mathcal{O}(N)$.

To get the final resource allocation result, the exhaustive searching scheme is used in RAVEC. The maximum number of matched combinations is $M*N$. The whole process should find out each possible combination to get the best result, so the computational complexity of RAVEC is $\mathcal{O}(MN)$.

4. SIMULATION RESULTS AND DISCUSSIONS

In this section, we will propose an actual example for RAVEC and verify RAVEC via simulations.

A. Scheme Stability

We assume that our mechanism is applied to a typical urban road scenario as depicted in Fig. 1. Vehicles move on a single lane and two direction road with RSUs deploying evenly along a straight road. We conduct several simulations to verify the effectiveness of RAVEC. We consider in a road section with a set of RSUs and a set of vehicles entering the road section in each time slot, with the assumption that the initial types of MEC-servers attached to RSUs follow a Gaussian distribution from 1 to K. MEC-servers work in an ideal environment. The velocity of each car follows a random distribution in a certain interval. Notable simulation parameters are summarized in Table 1.

Table 1. Parameters.

Parameter	Value
Number of vehicles in a time slot	20-30
Number of RSUs in a road section	12
Data size of vehicle's task	100-200 Mb
Computation size of vehicle's task	100-400 Mb
Computing resources of MEC servers(Upper bound)	1.2GHz
Velocity of vehicles	10-20m/s
Delay constraint	2s
Vertical distance between RSU and the center of the road	6m
Radius of RSU's communication coverage	200m
The interval between RSUs	300m
Transmission power of onboard units	1.3W
Bandwidth between vehicle and RSU	20MHz
Noise power	-110dB
Path loss exponent	2
Uplink channel fading coefficient	1.05

We simulate and evaluate the proposed task offloading mechanism via Matlab R2018b, the hardware environment is in Windows 7 with a 8 GB random access memory and an Intel Core i5 3.2GHz CPU.

Here, we define the probability of each MEC server attached to RSU being used in a time slot t as a set $M_t\{\mu_1, \mu_2, ..., \mu_n, ..., \mu_N\}$. To verify whether RAVEC can evenly select

the server along the road section, we calculate the variance of μ_n in set M_t through the formula below. The result is shown in Fig. 3. We can see that the variance in different time slots are stable, which shows the stability of RAVEC.

$$\begin{cases} \overline{\mu_t} = (\mu_1 + \mu_2 + ... + \mu_n + ... + \mu_N)/N \\ \sigma^2_{\mu_t} = ((\mu_1 - \overline{\mu_t})^2 + (\mu_2 - \overline{\mu_t})^2 + ... + (\mu_N - \overline{\mu_t})^2)/N \end{cases} \tag{12}$$

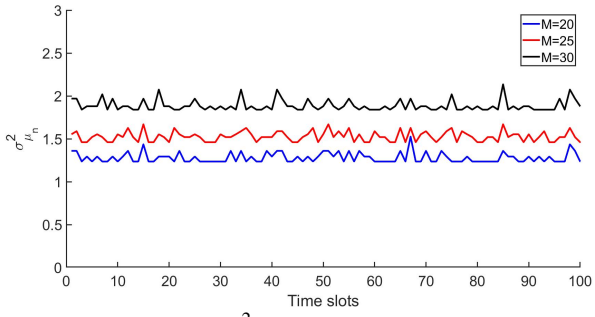
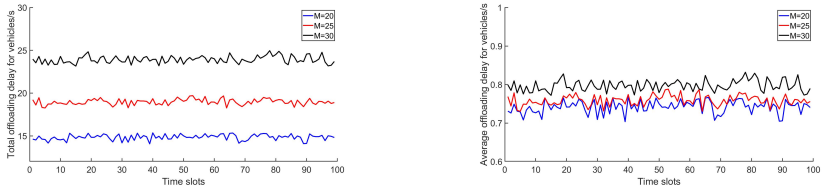


Fig. 3. $\sigma^2_{\mu_t}$ in different time slots.

In order to further test the stability of RAVEC, we simulate the offloading delay of different vehicle sets in multiple time slots. Each slot represents the total time in which the algorithm takes to complete a resource allocation process. From Figs. 4 (a) and (b) we can see that in different time slots, the average offloading delay of each vehicle is between 0.700 to 0.760 seconds when the number of vehicles is 20 and 0.740 to 0.790 seconds when the number of vehicles is 25, 0.770 to 0.840 seconds when the number of vehicles is 30. As the number of vehicles increase, the average delay also becomes larger. The reason for this relationship is that the total computation size of offloading tasks goes up as the number of vehicles rises which implies more resources of the MEC servers may be occupied. Our scheme considers that computing capacity of each MEC server changes in each time slot and proposes a MEC type model that follows the fact that the average time delay does not particularly increase as the number of vehicles increases and is relatively stable in different time slots.



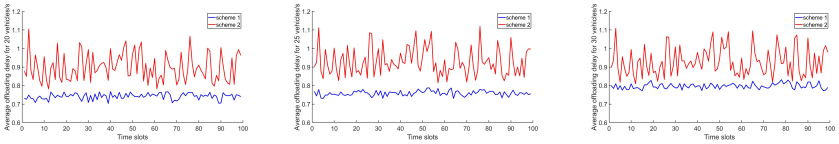
(a) Total offloading delay in different time slots. (b) Average offloading delay in different time slots.

Fig. 4. Delay performance.

B. Delay Performance Comparison

We evaluate RAVEC with an existing MEC offloading scheme that we conveniently named as the conventional scheme. In the conventional scheme, VUs consider to offload their onboard tasks to the nearest MEC server attached to the RSU by one hop without considering resource allocation among the whole set of vehicles in a time slot.

We compared the proposed scheme with the conventional scheme by fixing the number of vehicles to 3 levels: 20, 25 and 30. The results are shown below in Fig. 5 with scheme 1 represents RAVEC and scheme 2 represents the conventional scheme. We can see from the results that the range of delay jitter of the conventional scheme is much larger than RAVEC with RAVEC getting a 15% delay reduction comparing to the conventional scheme. RAVEC also makes better uses of the limited computation resources because of the comprehensive optimization algorithm.



(a) M=20 (b) M=25 (c) M=30

Fig. 5. Performance comparison.

5. CONCLUSIONS

In this paper, we investigated how to utilize the limited computation resources in MEC-enabled vehicular networks and proposed an optimal resource allocation mechanism named RAVEC. A MEC server type dividing mechanism was designed to optimize the allocation of computing resources while a total offloading delay minimum algorithm was developed to minimize the interaction during task offloading between vehicles. The location of vehicles and the occupying frequency of the target MEC server are also taken into consideration. Numerical results proved that RAVEC achieves an optimal way to offload the onboard task.

In summary, the major characteristic of RAVEC is to reduce the task processing latency of each vehicle. The transmission delay during the task offloading such as the wireless transmission delay between vehicle and RSU and the wire transmission delay between RSUs should also be considered. In different time slots, the mechanism can avoid significant changes of task offloading delay in order to keep the stability of the whole system. Beyond that, it can coexist with the legacy MEC-enabled offloading mechanisms in vehicular networks, and require only less changes to infrastructure, so that it can be easy to deploy incrementally.

In future works, we will investigate a scenario where the channel states and traffic states are more complicated, and improve the performance of RAVEC.

ACKNOWLEDGMENT

This paper is supported by National Key Research and Development Project of China (No. 2019YFF0303101).

REFERENCES

1. S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, “A comprehensive survey on vehicular ad hoc network,” *Journal of Network and Computer Applications*, Vol. 37, pp. 380-392.

2. "Analysis of event-driven warning message propagation in vehicular ad hoc networks," *Ad Hoc Networks*, Vol. 55, 2017, pp. 87-96.
3. 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 Communications Magazine*, Vol. 55, 2017, pp. 94-100.
4. J. Wan, L. Jianqi, S. Zehui, V. Athanasios, I. Muhammad, and Z. Keliang, "Mobile crowd sensing for traffic prediction in internet of vehicles," *Sensors*, Vol. 16, pp. 88-103.
5. Z. Zhou, P. Liu, Z. Chang, C. Xu, and Y. Zhang, "Energy-efficient workload offloading and power control in vehicular edge computing," in *Proceedings of IEEE Wireless Communications and Networking Conference Workshops*, 2018, pp. 191-196.
6. T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck, "Mobile edge computing potential in making cities smarter," *IEEE Communications Magazine*, Vol. 55, 2017, pp. 38-43.
7. N. Wang, B. Varghese, M. Matthaiou, and D. S. Nikolopoulos, "Enorm: A framework for edge node resource management," *IEEE Transactions on Services Computing*, 2017, p. 1.
8. Z. Zhou, H. Liao, B. Gu, K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "Robust mobile crowd sensing: When deep learning meets edge computing," *IEEE Network*, Vol. 32, 2018, pp. 54-60.
9. H. Zhou, H. Wang, X. Chen, X. Li, and S. Xu, "Data offloading techniques through vehicular ad hoc networks: A survey," *IEEE Access*, Vol. 6, 2018, pp. 65 250-65 259.
10. C. Huang, Y. Chen, S. Xu, and H. Zhou, "The vehicular social network (vsn)-based sharing of downloaded geo data using the credit-based clustering scheme," *IEEE Access*, Vol. 6, 2018, pp. 58 254-58 271.
11. K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Vehicular Technology Magazine*, Vol. 12, 2017, pp. 36-44.
12. K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proceedings of IEEE International Conference on Communications*, 2017, pp. 1-6.
13. H. Guo, J. Zhang, and J. Liu, "Fiwi-enhanced vehicular edge computing networks: Collaborative task offloading," *IEEE Vehicular Technology Magazine*, Vol. 14, 2019, pp. 45-53.
14. X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Transactions on Vehicular Technology*, Vol. 66, 2017, pp. 3435-3447.
15. T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Transactions on Communications*, Vol. 65, 2017, pp. 3571-3584.
16. J. Ren, G. Yu, Y. Cai, Y. He, and F. Qu, "Partial offloading for latency minimization in mobile-edge computing," in *Proceedings of IEEE Global Communications Conference*, 2017, pp. 1-6.
17. J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, Vol. 17, 2018, pp. 5506-5519.
18. X. Ma, S. Zhang, W. Li, P. Zhang, C. Lin, and X. Shen, "Cost-efficient workload scheduling in cloud assisted mobile edge computing," in *Proceedings of IEEE/ACM 25th International Symposium on Quality of Service*, 2017, pp. 1-10.

19. H. Guo and J. Liu, "Collaborative computation offloading for multiaccess edge computing over fiber–wireless networks," *IEEE Transactions on Vehicular Technology*, Vol. 67, 2018, pp. 4514-4526.
20. 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, 2018, pp. 22 622-22 633.
21. J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Transactions on Communications*, Vol. 66, 2018, pp. 1594-1608.
22. Q. Yuan, H. Zhou, J. Li, Z. Liu, F. Yang, and X. S. Shen, "Toward efficient content delivery for automated driving services: An edge computing solution," *IEEE Network*, Vol. 32, 2018, pp. 80-86.
23. K. Wang, H. Yin, W. Quan, and G. Min, "Enabling collaborative edge computing for software defined vehicular networks," *IEEE Network*, Vol. 32, 2018, pp. 112-117.
24. Y. Liu, S. Wang, J. Huang, and F. Yang, "A computation offloading algorithm based on game theory for vehicular edge networks," in *Proceedings of IEEE International Conference on Communications*, 2018, pp. 1-6.
25. J. Feng, Z. Liu, C. Wu, and Y. Ji, "Ave: Autonomous vehicular edge computing framework with aco-based scheduling," *IEEE Transactions on Vehicular Technology*, Vol. 66, 2017, pp. 10 660-10 675.
26. K. Zhang, Y. Mao, S. Leng, A. Vinel, and Y. Zhang, "Delay constrained offloading for mobile edge computing in cloud-enabled vehicular networks," in *Proceedings of the 8th International Workshop on Resilient Networks Design and Modeling*, 2016, pp. 288-294.
27. 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 Transactions on Wireless Communications*, Vol. 15, 2016, pp. 7272-7285.
28. G. Nawaz Ali, P. H. J. Chong, S. K. Samantha, and E. Chan, "Efficient data dissemination in cooperative multi-rsu vehicular ad hoc networks (vanets)," *Journal of Systems and Software*, Vol. 117, 2016, pp. 508-527.
29. D. Mazza, D. Tarchi, and G. E. Corazza, "A partial offloading technique for wireless mobile cloud computing in smart cities," in *Proceedings of European Conference on Networks and Communications*, 2014, pp. 1-5.

Gao-Feng Hong was born in Nanchang, Jiangxi province in 1995 and persuits Ph.D. at Beijing Jiaotong University. He is mainly engaged in the research of the vehicular network and mobile edge cloud computing.

Wei Su was born in October 1978. He received the Ph.D. degree in Communication and Information Systems from Beijing Jiaotong University in January 2008. Now he is a teacher in the School of Electronics and Information Engineering, Beijing Jiaotong University. He granted the title of Professor in November 2015. Dr. Su is mainly engaged in researching key theories and technologies for the next generation Internet and has taken part in many national projects such as National Basic Research Program(also called 973 Program), the Projects of Development Plan of the State High Technology Research, the National Natural Science Foundation of China. He currently presides over the research project "Fundamental Research on Cognitive Services and Routing of Future Internet", a project funded by the National Natural Science Foundation of China.

Qi-Li Wen was born in Qinghai province in 1995 and earned her bachelor of Communication Engineering degree in 2017. Now she is a postgraduate student at Beijing Jiaotong University. Her main research direction is the new generation of information network.

Peng-Lei Wu was born in Taizhou, Jiangsu province in 1993 and pursuits an MS degree at Beijing Jiaotong University. He is mainly engaged in the research of the control of multi-agent system.