J Inf Process Syst, Vol.17, No.3, pp.489~504, June 2021
https://doi.org/10.3745/JIPS.01.0077

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

An Offloading Scheduling Strategy with Minimized Power Overhead for Internet of Vehicles Based on Mobile Edge Computing

Bo He* and Tianzhang Li**

Abstract

By distributing computing tasks among devices at the edge of networks, edge computing uses virtualization, distributed computing and parallel computing technologies to enable users dynamically obtain computing power, storage space and other services as needed. Applying edge computing architectures to Internet of Vehicles can effectively alleviate the contradiction among the large amount of computing, low delayed vehicle applications, and the limited and uneven resource distribution of vehicles. In this paper, a predictive offloading strategy based on the MEC load state is proposed, which not only considers reducing the delay of calculation results by the RSU multi-hop backhaul, but also reduces the queuing time of tasks at MEC servers. Firstly, the delay factor and the energy consumption factor are introduced according to the characteristics of tasks, and the cost of local execution and offloading to MEC servers for execution are defined. Then, from the perspective of vehicles, the delay preference factor and the energy consumption preference factor are introduced to define the cost of executing a computing task for another computing task. Furthermore, a mathematical optimization model for minimizing the power overhead is constructed with the constraints of time delay and power consumption. Additionally, the simulated annealing algorithm is utilized to solve the optimization model. The simulation results show that this strategy can effectively reduce the system power consumption by shortening the task execution delay. Finally, we can choose whether to offload computing tasks to MEC server for execution according to the size of two costs. This strategy not only meets the requirements of time delay and energy consumption, but also ensures the lowest cost.

Keywords

Internet of Vehicles, Minimizing Power Overhead, Mobile Edge Computing, Optimization Model, Simulated Annealing Algorithm, Task Offloading

1. Introduction

At present, the automotive industry at home and abroad generally believes that low carbonization, informatization and intelligence are the future development directions of automobiles [1-3]. With this overall direction and trend, Internet of Vehicles came into being, providing the basic communication technology and the platform for in-vehicle applications. Introducing the mobile edge cloud computing into Internet of Vehicles can effectively solve the shortage of computing resources and storage resources of vehicles [4]. On this basis, through the research on the algorithm for offloading computing in Internet

Manuscript received July 27, 2020; first revision September 15, 2020; accepted October 4, 2020.
Corresponding Author: Bo He (hebohb123@126.com)
* College of Information Engineering, Guangzhou Institute of Technology, Guangzhou, China (hebohb123@126.com)
** Library, Jinan University, Guangzhou, China (ttianzli@jnu.edu.cn)

of Vehicles, the computing pressure of in-vehicle applications and services can be effectively alleviated, which makes the service delay as short as possible.

In [5], the author defined tasks with resource requirements and the maximum tolerable delay from the perspective of multi-access edge computing (MEC) service providers. Two algorithms are designed and simulated considering the service contract between the MEC service providers and vehicles, including resource provision and service price. They aimed to design optimized service contracts to increase the utilization rate of MEC servers while meeting vehicle service needs. In [6], the author defined a one-dimensional model, in which each MEC server had different computing capabilities and coverage. Taking the calculation time and the transmission time into consideration, the optimal multilevel offloading scheme is designed, using the Stackelberg game method. In addition, an iterative distributed algorithm is proposed and its convergence is proved. In reference [7], the author proposed the concept of quality of experience (QoE). It defined the utility function to reduce latencies and service costs from the perspective of car terminals. A distributed computing offloading algorithm was proposed and its superiority was proved by simulation. Huang et al. [8] studied the V2V offloading of mobile edge cloud computing based on software-defined networks in vehicle-mounted self-organizing networks. They specifically studied the communication topology between vehicles for task offloading. In [9], the researchers also studied task offloading and resource allocation based on mobile edge cloud computing in heterogeneous vehicle networks. Combined with the LTE unlicensed spectrum technology, the Q-learning distributed learning algorithm was used for channel and power allocation. Qiao et al. [10] proposed a concept of cooperative vehicle edge multi-access network. For immersive in-vehicle applications, it can reduce the perception reaction time and improve the driving safety and convenience. Du et al. [11] optimized the algorithm from two aspects: the vehicle terminal and the MEC/RSU side. In order to minimize the average cost and the cost of vehicle-mounted terminals and MEC/RSU, the author used the Lyapunov optimization theory to propose a low-complexity online algorithm for solving both problems in a comprehensive framework.

According to the status quo in current research, no matter from the perspective of MEC service providers or automotive terminals, most of the existing works consider the calculation delay and the communication delay, and then defines utility functions, designs a distributed or centralized offloading algorithm, and obtains an optimal solution. There is hardly any work that takes the energy consumption into consideration, and explores the multi-hop transmission cost between RSU and V2V. In 5G communications, the energy efficiency is a key point [12]. Due to the high device access density and data density in the 5G era, energy consumption will also greatly increase.

This paper proposes an offloading scheduling strategy for minimizing power overheads based on the mobile edge computing in Internet of Vehicles. The strategy fully considers the energy consumption problem in its design, and discusses the multi-hop transmission costs between RSU and V2V. Finally, setting the corresponding Monte Carlo simulation parameters and conditions, it is proved that the proposed strategy not only meets the delay and energy consumption requirements at high speeds and high densities, but also ensures the lowest cost.

2. System Model and Problem Modeling

2.1 System Model

The system model is shown in Fig. 1. RSUs are equidistantly distributed and the distribution interval

is L. Similarly, the access coverage diameter of each RSU is also L. Therefore, a road can be divided into many sections according to the coverage of RSU, and the length of each section is L. In addition, each vehicle only communicates with the RSUs that it has access to, and includes the uplink data transmission and the downlink data download. In this model, each RSU is connected to a MEC server via a wired cable. It can be considered that the communication bandwidth is sufficiently large; thus, the transmission time between RSU and MEC is negligible.

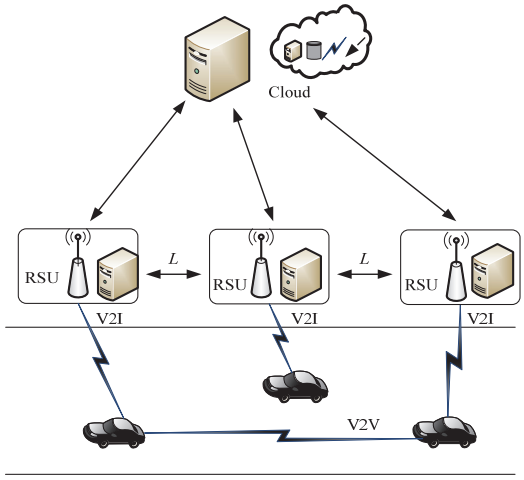

Fig. 1. System model.

The distribution of vehicles follows a one-dimensional Poisson point process with a density λ along the road. In other words, the distance between two adjacent vehicles is independent, and it follows the exponential distribution with parameter λ. The probability density function of the distance l between any two adjacent vehicles is as follows:

$$f_d(l) = \lambda \cdot e^{-\lambda d} \quad (l > 0) \tag{1}$$

X is the distance from the rear boundary of RSU coverage when vehicles initiate the calculation offload to the MEC server. After the computing task offload is initiated, the stay time t in current RSU coverage is:

$$t = \frac{L - X}{v} \tag{2}$$

Next, the description of a computing task is considered. For a certain type of on-board computing task i, it will be denoted by $F_i(c_i, d_i, t_{i,max})$, where c_i is the amount of calculation and may specifically be the number of CPU operation cycles required for computing tasks. d_i is the amount of input data required for calculation, and $t_{i,max}$ is the maximum tolerable delay of computing tasks. Assuming that there are I tasks, and the proportion of task i is ε_i, then:

$$\sum_{i=1}^{I} \varepsilon_i = 1 \tag{3}$$

The vehicle that initiates computing task i is referred to as a type i vehicle, and its speed is v. Let c_v denote the computing power of vehicles (which can specifically be the CPU frequency, i.e., the number of computing cycles executed by CPU in one second), and c_m denote the computing power of MEC servers. R_i is the type i vehicle uplink communication rate, i.e., the rate at which the vehicle uploads data to its connected RSU. Generally speaking, the amount of input data of a computing task is much larger than the amount of output data, such as the virtual reality and augmented reality applications. Thus, in the study of this paper, the download delay of calculation output is ignored, and only the multi-hop transmission delay of computing results is considered.

2.2 Model for Calculation of Offload Energy Consumption

This paper primarily analyzes the local execution time of computing tasks and the execution time offloaded to the MEC server from the perspective of latency. Considering that the vehicle speed is too high, and the computing task or the MEC server load is heavy, the computing output needs to be transmitted by multi-hop RSUs. A predictive offloading scheduling algorithm based on the MEC server load status is designed. $F_i(c_i, d_i, t_{i,max})$ is used to define an on-board computing task. In fact, $t_{i,max}$ is not used in the subsequent derivation and analysis. As defined, $t_{i,max}$ represents the maximum tolerable delay. As long as the calculated service completion time is less than this value, the service quality can be accepted by vehicle users. On this basis, further considering the energy consumption of locally performing computing tasks or offloading to the MEC server, the delay and the energy consumption are better balanced. According to the study [13], the calculated energy consumption (power) and computing power c (CPU frequency) satisfy the following relationship:

$$P = k \cdot c^{\alpha} \tag{4}$$

where, k and α are parameters. According to the study [14], $\alpha = 3$ will be taken. When computing tasks are performed locally, there is no communication process; thus, only the calculation power consumption is considered. When task $F_i(c_i, d_i, t_{i,max})$ is executed locally, the energy consumption is $W_{i,local}$, then:

$$W_{i,local} = P_{local} \cdot t_{i,local} = k \cdot c_v^{\alpha} \cdot \frac{c_i}{c_v} = k \cdot c_v^{\alpha-1} \cdot c_i = k c_i c_v^2 \tag{5}$$

where P_{local} is the local calculated CPU power.When the vehicle needs to offload computing tasks to MEC servers for execution, it directly uploads the computing input data to the currently connected RSU via V2I. In other parlance, the computing task is offloaded to MEC servers to which the RSU is connected. In this case, it is foreseeable that if the vehicle speed is very high, or the calculation amount of computing tasks is large, much calculation time is required. The computing result needs to be transmitted back to vehicles through multi-hop RSUs, and the communication between RSUs needs to go through the wireless backhaul link. Moreover, the backhaul link is relatively unstable, with large fluctuations and high latencies. The entire process is shown in Fig. 2.

For computing tasks performed on the MEC server, the energy consumption is mainly composed of three parts. The communication energy consumption $W_{i,upload}$ of uploading data to the RSU connected to the MEC server, the calculation energy consumption $W_{i,compute}$ on the MEC server, and the energy consumption $W_{i,download}$ of the multi-hop back transmission calculation output. The total energy

consumption of the system where the computing task $F_i(c_i, d_i, t_{i,max})$ offloaded to the MEC server is recorded as $W_{i,MEC,SYS}$, then:

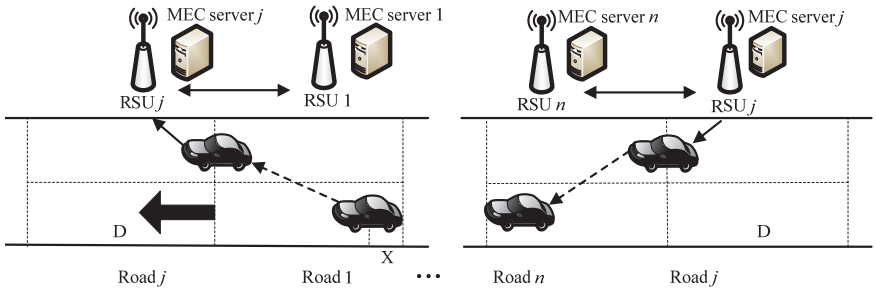

Fig. 2. Offloading model.

$$W_{i,MEC,SYS} = W_{i,upload} + W_{i,compute} + W_{i,download} = p_i \cdot \frac{d_i}{R_i} + kc_i c_m^2 + W_0 \cdot x_i \tag{6}$$

where W_0 is the energy consumption of I2I transmission data between RSUs covered by a section of RSU or by the energy consumption of V2V transmission data in a section of RSU, corresponding to different strategies in Section 3. p_i is the communication power for uploading data, i.e., the transmission power of the vehicle antenna. R_i is the data uploading rate. According to Shannon's formula, the relationship between p_i and R_i is:

$$R_i = B \log_2 \left(1 + \frac{p_i H_i}{\sigma^2}\right) \tag{7}$$

where B is the channel bandwidth for uploading data to the RSU, H_i is the channel gain from vehicle i to the RSU, and σ^2 is the channel noise power. In fact, from the perspective of the vehicle, there is no need to consider the calculated energy consumption $W_{i,compute}$ of the MEC server and the energy consumption of calculation data backhaul. Thus, we only consider the uploading energy consumption in the next step, namely:

$$W_{i,MEC} = W_{i,upload} = p_i \cdot \frac{d_i}{R_i} \tag{8}$$

We define the cost of executing a computing task $F_i(c_i, d_i, t_{i,max})$ as $COST_i$. Define $\delta_{i,t}$ and $\delta_{i,W}$ as the time preference factor and the energy consumption preference factor of computing tasks, then:

$$COST_i = \delta_{i,t} t_i + \delta_{i,W} W_i \tag{9}$$

usually, $0 \le \delta_{i,t} \le 1$, $0 \le \delta_{i,W} \le 1$. Then, for computation task $F_i(c_i, d_i, t_{i,max})$, the cost of its local execution is:

$$COST_{i,local} = \delta_{i,t} t_{i,local} + \delta_{i,W} W_{i,local} = \delta_{i,t} \frac{c_i}{c_v} + \delta_{i,W} kc_i c_v^2 \tag{10}$$

The cost of execution on the MEC server is:

$$COST_{i,MEC} = \delta_{i,t} t_{i,MEC} + \delta_{i,W} W_{i,MEC} = \delta_{i,t} \left[\frac{d_i}{R_i} + \frac{c_i}{c_m} + (t_0 \times x_i)\right] + \delta_{i,W} \cdot p_i \cdot \frac{d_i}{R_i} \tag{11}$$

For computing task $F_i(c_i, d_i, t_{i,max})$, we define ξ_i as the offload option. $\xi_i = 1$ means to choose to execute locally, $\xi_i = 0$ means to choose to offload the computing task to the MEC server to perform calculation. Therefore, the cost of the entire task can be expressed as:

$$COST_i = \xi_i COST_{i,local} + (1 - \xi_i) COST_{i,MEC} \tag{12}$$

For the entire system of Internet of Vehicles that introduces the mobile edge cloud computing, the total cost can be written as:

$$COST_{ALL} = \sum_{i=1}^{S} \varepsilon_i COST_i \varepsilon_i \tag{13}$$

where ε_i is the proportion of computing task $F_i(c_i, d_i, t_{i,max})$ in all tasks. Next, we will optimize the computing task and the entire system based on these expressions.

2.3 Analysis for Calculating Offload Delay

For vehicles, the execution of computing tasks is divided into two cases: local execution and offloading to MEC servers for execution. When the computing task is selected to be performed locally, the calculation input data is located in the vehicle storage device. There is no additional communication overhead required in the calculation process; thus, only the calculation time needs to be considered. For a type of task, the completion delay is only related to the computing power c_v of the vehicle. For type *i* vehicles that perform type *i* tasks, it is noted that the local execution time is $t_{i,local}$, then:

$$t_{i,local} = \frac{c_i}{c_v} \tag{14}$$

The execution time of the MEC server is mainly composed of three parts: the calculation data uploading time, the MEC server execution calculation time and the data return time, which are respectively denoted as $t_{i,upload}$, $t_{i,compute}$, and $t_{i,download}$. In most computing applications, the amount of calculated output data is much smaller than the amount of input data. Therefore, the transmission time from the RSU to vehicles is not considered. Thus, $t_{i,download}$ is only composed of the multi-hop transmission time taken by the vehicle to pass through multiple RSU coverage areas due to the excessively high vehicle speed or the long calculation time. For type *i* vehicles that perform type *i* tasks, the execution time to be offloaded to the MEC server is $t_{i,MEC}$, then:

$$t_{i,MEC} = t_{i,upload} + t_{i,compute} + t_{i,download} = \frac{d_i}{R_i} + \frac{c_i}{c_m} + (t_0 \times x_i) \tag{15}$$

When the speed of vehicles is too high or the calculation time required for tasks is quite long, after the computing task offloaded to the MEC server is completed, the vehicle has left the access range of RSU connected to the MEC server responsible for calculation, and multi-hop transmission is required to transfer the results back to vehicles. Where x_i is the number of hops and t_0 is the transmission time of a single hop. Combined with the residence time of vehicles in the coverage of RSU connected to the MEC server that initiated computing tasks and offloading, the expression that can be derived for x_i is as follows, where ⌈ ⌉ means to round up.

$$x_i = \begin{cases} \left\lceil \frac{v_i}{D} \times \left(\frac{d_i}{R_i} + \frac{c_i}{c_m} - \frac{D-X}{v_i} \right) \right\rceil & \frac{d_i}{R_i} + \frac{c_i}{c_m} - \frac{D-X}{v_i} > 0 \\ 0 & \frac{d_i}{R_i} + \frac{c_i}{c_m} - \frac{D-X}{v_i} \le 0 \end{cases} \tag{16}$$

2.4 Optimization Model for Offloading

In order to minimize the cost of local execution, we can choose an optimal CPU operating frequency within the actual range, so that both the delay and the energy consumption meet requirements, and the cost is thus minimized. The optimization problem can be expressed as:

$$\begin{aligned} &\min_{c_v} g\left(c_v\right) = \delta_{i,t} \frac{c_i}{c_v} + \delta_{i,W} k c_i c_v^2 \\ &s.t. C1: t_{i,local} = \frac{c_i}{c_v} \le t_{i,\max} \\ &\quad C2: W_{i,local} = k c_i c_v^2 \le W_{i,\max} \\ &\quad C3: C_{v,\min} \le c_v \le C_{v,\max} \end{aligned} \tag{17}$$

where $W_{i,max}$ is the maximum tolerable energy consumption of type i tasks, $C_{v,min}$ and $C_{v,max}$ are the minimum and the maximum operating frequencies of the vehicle's local CPU respectively.

3. Simulated Annealing Algorithm for Solving the Optimization Model

Due to the nonlinear relationship between the optimization goal and the constraints mentioned above, the optimization mode is difficult to solve. Therefore, the simulated annealing algorithm is used to solve this optimization model.

3.1 Establishment of Metropolis Guidelines

Set in the current state i, the solution of system is W_i, and the solution in the next state j is W_j. According to the metropolis criterion, the following formula holds:

$$W_i - W_j = \begin{cases} > 0 & \textit{Accept the current solution as the optimal solution} \\ < 0 & \textit{Accept the current solution as the optimal solution with probability } p \end{cases} \tag{18}$$

where p is the probability of accepting the current solution as the optimal solution, which can be expressed as:

$$p = \begin{cases} 1 & W_i > W_j \\ \exp\left(\frac{-\left(W_i - W_j\right)}{\varphi T} \right) & W_i < W_j \end{cases} \tag{19}$$

where φ is the Boltzmann's constant.

3.2 Algorithm Process

The algorithm process is shown in Algorithm 1.

Algorithm 1. Simulated annealing algorithm

1. **Initialization:** high temperature $T_{max} = 100$, lower temperature limit $T_{min} = 1$;
2. Set the current state to the highest temperature, i.e., $T(k) = 100$;
3. Initialize the first solution state $S_1 = \{\delta_{i,t}^1, \delta_{i,W}^1\}$ and calculate the corresponding W_i^1;
4. Assign $S_1 = \{\delta_{i,t}^1, \delta_{i,W}^1\}$ to the best solution $S^* = \{\delta_{i,t}^*, \delta_{i,W}^*\}$;
5. Initialize the number of iterations at the current temperature $L = 100$;
6. **For** $m = 1: L$ do
   Dither to solve the new state $S' = \{\delta_{i,t}', \delta_{i,W}'\}$, and calculate the corresponding W_i';
7. Calculate the increment $\Delta W_i = W_i' - W_i^1$;
   If $\Delta W_i < 0$ then
   Accept the current solution as the optimal solution and assign $S_1 = \{\delta_{i,t}^1, \delta_{i,W}^1\}$ to the optimal solution $S^* = \{\delta_{i,t}^*, \delta_{i,W}^*\}$;
   Else
   Accept $S' = \{\delta_{i,t}', \delta_{i,W}'\}$ as the current optimal solution with probability p in metropolis criterion
   End if
   If $m = L$
   End if
   Else return 6
8. $T(k+1) \le \alpha T(k)$
   If $T(k+1) \le T_{min}$
   End if
   Else return 6

4. Simulation Experiment and Result Analysis

In this section, we will select multiple types of computing tasks, including the computationally intensive, the delay sensitive, etc., and compare our proposed strategy with the two methods in [9] and [11] to verify our strategy. Taking the vehicle speed and the vehicle density as variables, the simulation calculates the delay and offloads the energy consumption. By the MATLAB simulation platform, and in accordance with the relevant provisions of the MEC white paper, a system model is built. The hardware environment used in the experiment is a Samsung laptop, specifically configured as Intel Core i5-7300HQ CPU @ 2.5GHz, 8G memory, 500G hard drive, Windows 10 operating system.

4.1 Parameter Settings

The computing tasks of five parameters are selected, which represent ordinary applications in normal applications, resource-intensive applications (both the amount of calculation and the amount of data are relatively large), delay-sensitive (the maximum tolerable time is short), calculation-intensive (a large amount of calculation) and data-intensive (a large amount of data). The specific data and the proportion of each task are shown in Table 1.

The parameters of these tasks have no actual units, and are only used to reflect the characteristics of tasks. On this basis, the simulation experiment sets the following parameters, as shown in Table 2.

Table 1. Computing task type and data

$F_i(c_i, d_i, t_{i,max})$	Task type	Ratio ε_i
F1(10,50,1)	Normal type	0.2
F2(200,200,5)	Resource-intensive	0.2
F3(5,5,0.5)	Delay-sensitive	0.2
F4(500,100,50)	Calculation-intensive	0.2
F5(100,500,10)	Data-intensive	0.2

Table 2. Simulation parameters

Parameter	Value
The computing power of vehicles c_v	15
The uplink communication rate of vehicles R_i	100
The computing power of MEC server c_m	50
RSU pitch L	100 m
The vehicle density per unit road section λ_u	0.2–10
The speed of vehicles v	40–120 km/hr
The RSU communication delay of unit road section t_o	0.5 s
Single set of parameters Monte Carlo simulation times	1000000
The V2V communication delay of unit road section $r_{V2V} \cdot t_o$	0.1 s

The unit of vehicle density λ_u per road section is not m^{-1}, but the average number of vehicles per RSU. By definition, when the RSU space is 100 m, the actual density is $\frac{\lambda_u}{100} m^{-1}$.

The simulation method for the queuing time of a single server will be explained below. For T_{W1}, this queuing process can first be regarded as a M/M/1 queuing process with a mortality rate, i.e., a task completion rate of $\mu = 1/\sum_{i=1}^{I} \varepsilon_I \left(\frac{c_i}{c_m}\right)$. During the simulation, a random number from 0 to 1 is generated. According to the discrete probability distribution of states, the state where the server is located can be simulated. Taking $\lambda = 0.2$ and $\mu = 0.5$ as examples, the probability that the server is in each state is shown in Table 3.

Table 3. Server state probability (λ=0.2, μ=0.5)

State	0	1	2	3	4	...
Probability	0.6	0.24	0.096	0.0384	0.01536	...
Random number	0–0.6	0.6–0.84	0.84–0.936	0.936–0.9744	0.9744–0.098976	...

Suppose the server is in state 0 ($T_{W1} = 0$). Assuming that the server is in state $k(k > 1)$, there are k – 1 tasks waiting for computing services. A task is receiving computing services. First, we simulate the remaining time of task receiving computing services and generate a random number from 0 to 1. According to the proportion of computing tasks, the types of computing tasks are simulated. Then, a random number from 0 to 1 is generate to simulate the completion progress of tasks that are receiving the calculation. T_{W1} can be expressed as follows:

$$T_{W1}\left(state=k\right)=\theta\cdot T_{w1}+T_{w2}+\cdots+T_{w(k-1)}+T_{wk} \tag{20}$$

where $T_{wk}(x=1,2,\ldots k)$ is an independent discrete random variable, representing the calculation time of each task in the queue. The probability distribution is shown in Table 4.

Table 4. Probability distribution of T_{wk}

	Task type				
	***F*1(10,50,1)**	***F*1(10,50,1)**	***F*1(10,50,1)**	***F*1(10,50,1)**	***F*1(10,50,1)**
Value	0.2	4	0.1	10	2
Probability	0.3	0.1	0.2	0.2	0.2

Moreover, θ is a random variable subject to the uniform distribution $U(0,1)$, and represents the progress of completing the computing task of receiving the MEC server. When the vehicle initiates a computing task, the distance X from the boundary of rear coverage of the currently connected RSU follows a uniform distribution $U(0,D)$.

4.2 Comparative Analysis of Delay Energy Consumption with Vehicle Density Changes

4.2.1 Average time with vehicle density changes for five computing tasks in three offloading strategies

The following will analyze the average time change of three offloading strategies with the change of vehicle density and speed per unit road section. Fig. 3 shows the average time of five types of computing tasks as a function of vehicle density. The selected vehicle speed is 70 km/hr.

It can be seen that when the five tasks under different vehicle densities use our proposed predictive V2V offload strategy based on the MEC load status, the average time of calculation is less than the previous two existing strategies. For normal tasks, it can be seen that the proposed strategy does not perform much differently from the strategy in [9]. As the calculation time is short, the MEC queuing time saved by the proposed strategy is not outstanding. When the vehicle density is low, the calculation results' return time of the strategy in [9] and the proposed strategy becomes evident. For resource-intensive computing tasks, we can see that when the vehicle density is between 5 and 7, the time saved by our proposed strategy is more prominent. As the vehicle density continues to increase, the total time spent performing calculations at the MEC may be longer than the time spent performing calculations locally. The offloading ratio approaches zero, and the time of three strategies will also approach. Since the local execution has been chosen in most cases, the time is same.

For delay-sensitive tasks, we can see that there is seldom any difference among these three strategies. As this task has a small amount of calculation, the local execution time is short. When executed on the MEC server, the task will introduce the data uploading time and possibly the multi-hop calculation result return time. For calculation-intensive tasks, it is predictable that the proposed strategy significantly saves the MEC server queuing time. As can be seen, when the vehicle density is extremely high, much time can still be saved. For data-intensive tasks, it can be seen that when the vehicle density is low, the strategy in [9] and the proposed strategy have greater advantages over the strategy in [11]. As the vehicle density increases, the advantages of our proposed offloading strategy become more apparent.

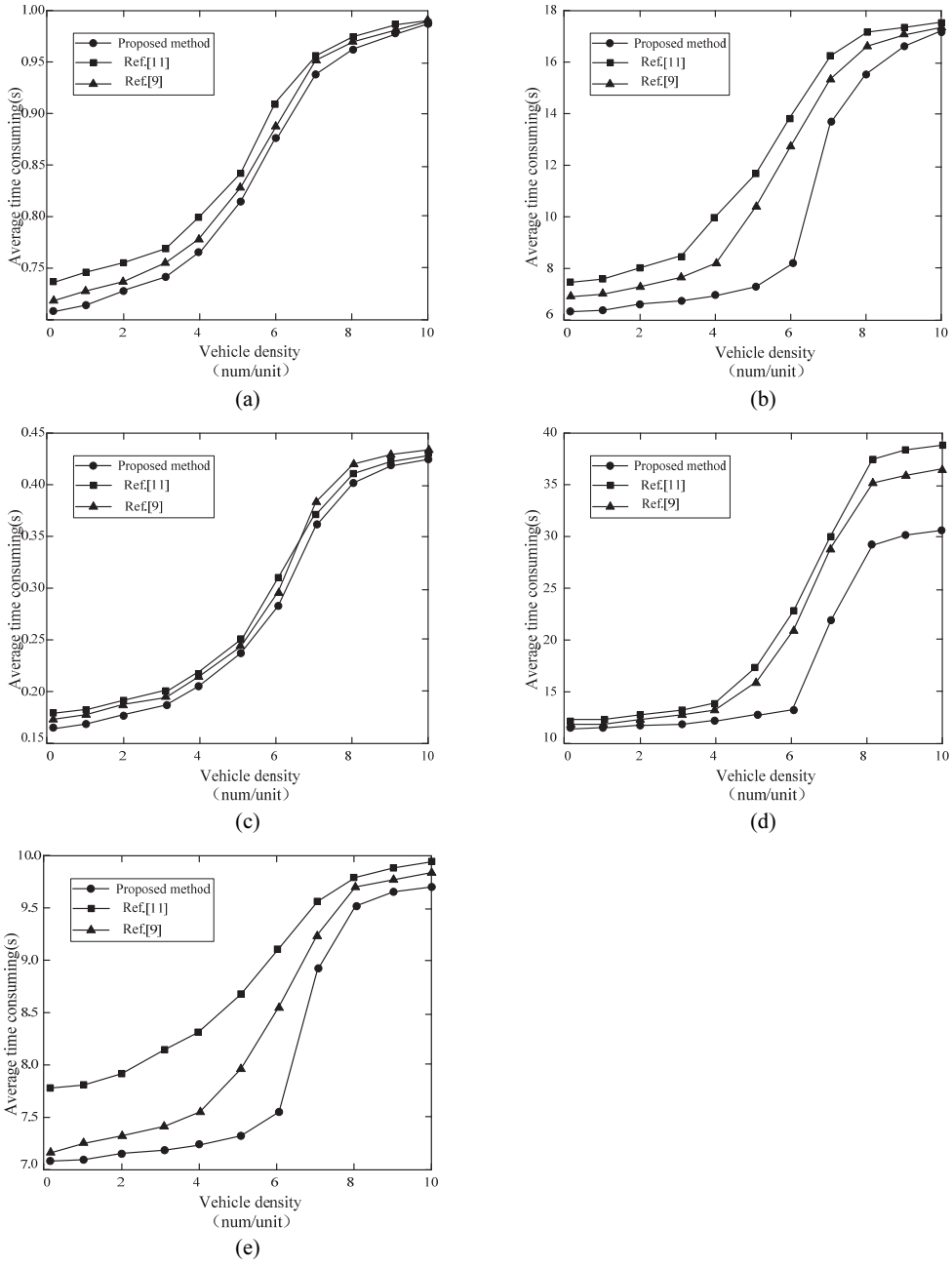

Fig. 3. Average time with vehicle density changes for five computing tasks in three offloading strategies: (a) Type1, (b) Type2, (c) Type3, (d) Type4, and (e) Type5.

4.2.2 Average energy consumption with vehicle density changes for five computing tasks in three offloading strategies

The results in Table 5 show the average energy consumption of three methods at different vehicle densities. The following will analyze the average energy consumption changes in three offloading strategies with the change of vehicle density and speed per unit road segment.

Table 5. Average energy consumption of three methods under different vehicle density (unit: J)

Group	Method	Vehicle density (number per unit)				
		2	4	6	8	10
Type1	Proposed	6.54	10.45	14.68	18.80	23.67
	Zhang et al. [9]	7.65	12.43	16.87	20.45	27.48
	Du et al. [11]	8.21	13.56	17.80	22.42	28.94
Type2	Proposed	4.42	6.04	8.68	12.64	16.82
	Zhang et al. [9]	4.87	6.98	9.76	13.90	18.03
	Du et al. [11]	5.04	7.33	10.34	14.63	19.25
Type3	Proposed	9.45	13.55	19.84	25.80	31.64
	Zhang et al. [9]	12.53	15.76	23.89	27.69	34.73
	Du et al. [11]	12.23	16.88	26.86	28.36	35.67
Type4	Proposed	0.85	1.56	2.03	2.66	3.02
	Zhang et al. [9]	0.89	1.90	2.12	2.87	3.31
	Du et al. [11]	1.02	1.95	2.21	2.97	3.45
Type5	Proposed	1.23	2.32	3.34	4.98	6.32
	Zhang et al. [9]	1.36	3.03	3.90	5.38	7.76
	Du et al. [11]	1.31	2.88	4.02	5.66	8.03

It can be seen from Table 5 that the total system energy consumption is decreasing with the increase in the delay constraint range. According to the analysis in Table 5, this is because as the range of delay constraint increases, more and more users choose the local computing. The greater the delay constraint, the more energy saved by the local computing. In addition, when the delay constraint range is small, there are many delay-sensitive users. For these users, due to the fact that there are more wireless resources available during offloading, the co-channel interference will be more severe as a result of frequent reuse. This will cause an increase in the system energy consumption. Thus, the total energy consumption of the system decreases with the increase in the delay constraint range.

4.3 Comparative Analysis of Delay Energy Consumption with Vehicle Speed Changes

4.3.1 Average time with vehicle speed changes for five computing tasks in three offloading strategies

Fig. 4 illustrates changes of average time with the vehicle speed for five computing tasks. The selected vehicle density is 4 (per segment RSU).

For normal computing tasks, it can be seen that the proposed predictive offloading algorithm based on the MEC load status is not significantly different from the offloading strategy in [9], which is superior to the offloading strategy in [11]. Due to the fact that the computational task is relatively less burdensome, when the vehicle speed is high (there are many RSU coverage sections passed during this time), our proposed strategy is superior to the offloading strategy in [9].

For resource-intensive computing tasks, it can be seen that when the vehicle speed is between 80 and

120 km/hr, the proposed strategy shows a greatly improved performance compared to the first two strategies. When the vehicle speed is high, $x_i = \left[\frac{v_i}{D} \times \left(\frac{d_i}{R_i} + \frac{c_i}{c_m} - \frac{D - X}{v_i} \right) \right]$ and x_i will increase, and there are more MEC servers for offloading. Therefore, a large amount of MEC server queuing time is saved. For delay-sensitive computing tasks, as shown in this figure, these three strategies have seldom any time difference, and most of them choose to execute locally. For calculation-intensive computing tasks, our proposed offloading strategy reduces the average time compared to the other two strategies. Furthermore, as the vehicle speed increases, the time increase becomes less apparent. For data-intensive computing tasks, when the vehicle speed is high, the average time is also greatly reduced.

4.3.2 Average energy consumption with vehicle speed changes for five computing tasks in three offloading strategies

The results in Table 6 show the average energy consumption of three methods at different vehicle speeds. The following will analyze the average energy consumption changes in three offloading strategies with the change of vehicle density and speed per unit road segment.

As can be seen from Table 6, the total energy consumption of the system is decreasing as the delay constraint range increases. According to the analysis in Table 6, the reason is that as the range of delay constraints increases, more and more users choose local computing. The greater the delay constraint, the more energy saved by local computing. In addition, when the delay constraint range is small, there are many delay-sensitive users. For these users, due to the large amount of wireless resources that are allocated during offloading, the co-frequency interference based on the frequency of reuse will be more severe, which will enhance the system energy consumption. Therefore, with the increase in the delay constraint range, the total system energy consumption is reduced.

Table 6. Average energy consumption of three methods at different vehicle speeds (unit: J)

Group	Method	Vehicle speed (km/hr)				
		2	4	6	8	10
Type1	Proposed	6.03	8.21	10.68	13.87	17.84
	Zhang et al. [9]	7.12	9.74	11.89	16.48	18.95
	Du et al. [11]	7.25	9.52	14.56	17.64	20.45
Type2	Proposed	3.89	5.35	8.95	12.84	17.48
	Zhang et al. [9]	4.31	6.78	9.78	14.78	19.48
	Du et al. [11]	5.08	7.02	10.37	16.37	21.05
Type3	Proposed	8.43	13.56	17.89	24.23	30.63
	Zhang et al. [9]	9.84	15.32	20.85	26.68	33.79
	Du et al. [11]	10.28	16.08	21.36	28.45	35.02
Type4	Proposed	1.35	2.53	3.13	4.66	6.26
	Zhang et al. [9]	1.66	3.08	3.85	5.67	7.06
	Du et al. [11]	1.91	3.23	4.09	5.94	7.83
Type5	Proposed	1.53	2.26	3.23	3.96	4.82
	Zhang et al. [9]	1.88	2.99	4.32	4.87	5.81
	Du et al. [11]	2.04	3.25	4.27	5.67	6.35

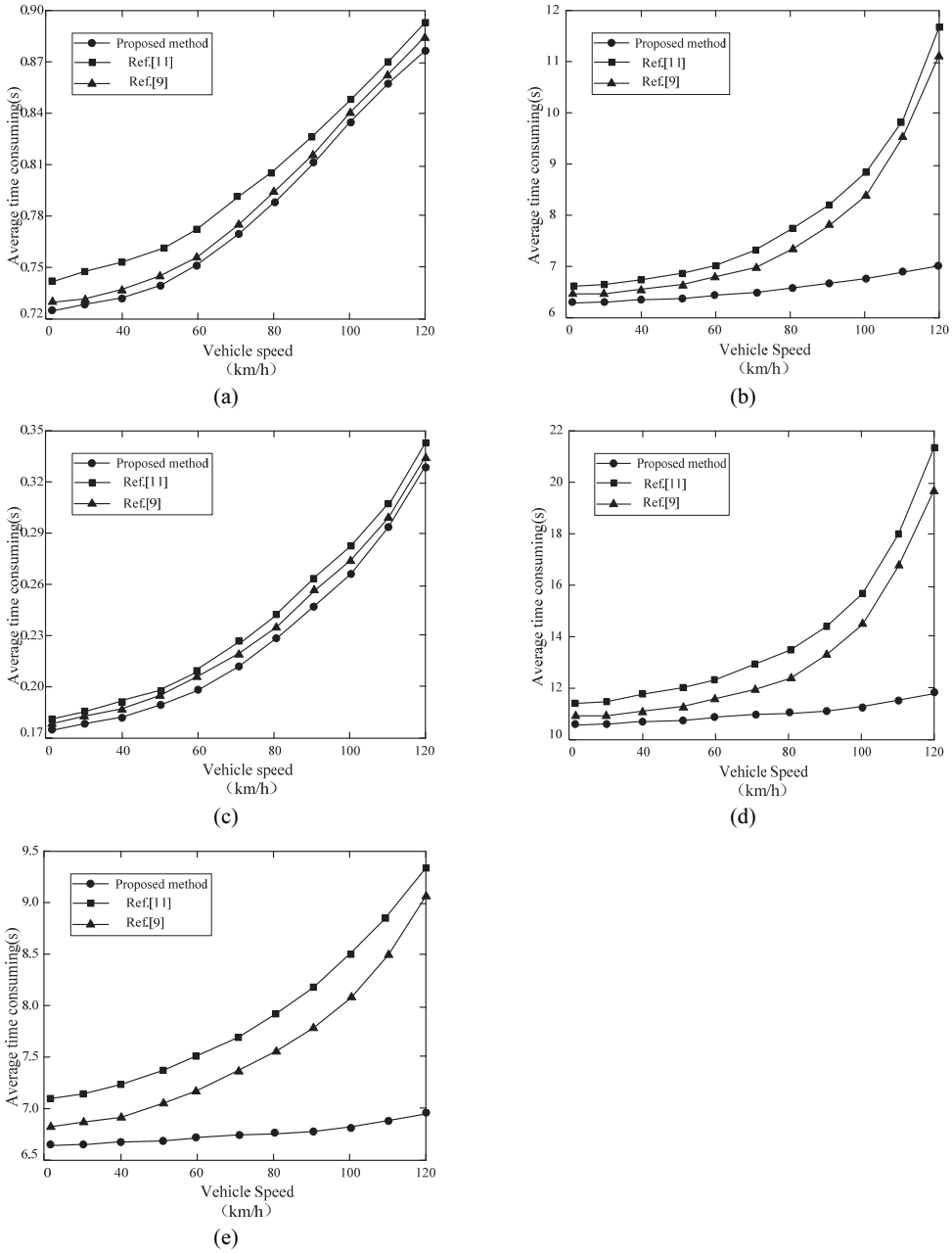

Fig. 4. Average time change with vehicle speed for five computing tasks in three offloading strategies: (a) Type1, (b) Type2, (c) Type3, (d) Type4, and (e) Type5.

5. Conclusion

Based on two existing offloading strategies, this paper proposes a predictive offloading algorithm based on the MEC load status. Predictability in our study refers to the estimation of task uploading and calculation time, the uploading of calculation input data by V2V communication, and the return of

calculation results. When the vehicle speed is too high or computing tasks are heavy, the data backhaul time of multi-hop RSU backhaul link is saved. In other words, based on the MEC status, during the above-mentioned period of time, multiple RSU coverage sections are passed. According to the MEC status information, we select the MEC server with the least waiting time to offload computing tasks. When the vehicle density is high or the vehicle speed is fast, the waiting time on the MEC server is saved. In this paper, a simple linear model is employed to discuss the delay and energy consumption, regardless of whether it is a simple analysis of delay or a comprehensive analysis of delay and energy efficiency. In fact, the user satisfaction cannot be described by a simple linear model in terms of individual's subjective feelings. User satisfaction models on delay and energy consumption can be introduced in the future work. As such research is more in line with subjective feelings, it is more applicable.

Acknowledgement

This work was supported by Characteristic Innovation Project from Guangdong Provincial Department of Education in 2019 (No. 2019gktscx073).

References

[1] Q. Liu, D. Xu, B. Jiang, and Y. Ren, "Prescribed-performance-based adaptive control for hybrid energy storage systems of battery and supercapacitor in electric vehicles," *International Journal of Innovative Computing, Information and Control*, vol. 16, no. 2, pp. 571-584, 2020.

[2] W. Xu, S. Wang, S. Yan, and J. He, "An efficient wideband spectrum sensing algorithm for unmanned aerial vehicle communication networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1768-1780, 2019.

[3] R. Hu, H. M. Zhao, and Y. Wu, "The methods of big data fusion and semantic collision detection in Internet of Thing," *Cluster Computing*, vol. 22, no. 4, pp. 8007-8015, 2019.

[4] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Mobile edge computing and networking for green and low-latency Internet of Things," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 39-45, 2018.

[5] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, Paris, France, 2017, pp. 1-6.

[6] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proceedings of 2017 IEEE International Conference on Communications (ICC)*, Paris, France, 2017, pp. 1-6.

[7] Q. Liu, Z. Su, and Y. Hui, "Computation offloading scheme to improve QoE in vehicular networks with mobile edge computing," in *Proceedings of 2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*, Hangzhou, China, 2018, pp. 1-5.

[8] C. M. Huang, M. S. Chiang, D. T. Dao, W. L. Su, S. 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, 2018.

[9] H. Zhang, Q. Luan, J. Zhu, and F. Li, "Task offloading and resource allocation in vehicle heterogeneous networks with MEC," *Chinese Journal on Internet of Things*, vol. 2, no. 3, pp. 36-43, 2018.

[10] G. Qiao, S. Leng, K. Zhang, and Y. He, "Collaborative task offloading in vehicular edge multi-access networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 48-54, 2018.

[11] J. Du, F. R. Yu, X. Chu, J. Feng, and G. Lu, "Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1079-1092, 2018.

[12] 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, no. 2, pp. 36-44, 2017.

[13] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proceedings of 2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, Boston, MA, 2019.

[14] J. Zhang, X. Hu, Z. Ning, E. C. H. Ngai, L. Zhou, J. Wei, J. Cheng, and B. Hu, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2633-2645, 2018.

Bo He **https://orcid.org/0000-0002-0077-9312**

She has a master of Software Engineering, She is a senior engineer. She graduated from Jinan University in 2001. She is working in Guangzhou Institute of Technology. Her research interests include software engineering and graphic image.

Tianzhang Li **https://orcid.org/0000-0002-6594-9431**

He has a master of Computer Science, He is a senior engineer. He graduated from Jinan University in 2013. He is working in Jinan University. His research interests include big data and Digital Library.