Task Offloading Based on Edge Collaboration in MEC-Enabled IoV Networks

Taoyu Deng, Yueyun Chen, Guang Chen, Meijie Yang, and Liping Du

***Abstract*—Benefiting from its abundant computing resources and low computing latency, mobile edge computing (MEC) is a promising approach for enhancing the computing capacity of the 5G Internet of vehicles (IoV). Because of the high mobility, handover is frequent and inevitable in IoV networks. In this paper, we investigate an edge collaborative task offloading and splitting strategy in MEC-enabled IoV networks, in which the task is splitted on the edge and paralleling executed by each part of the task on several MEC servers when handover is occured. Applications in IoV networks have flexible requirements on latency and energy consumption. To realize the tradeoff between latency and energy consumption, we formulate the task offloading and splitting as an optimization problem with the aim of minimizing the total cost of latency and energy consumption by jointly optimizing the task splitting ratio and uplink transmit power of vehicle terminal (VT). Because the proposed problem is non-smooth and non-convex, we divide the original problem into two convex subproblems, and apply an alternate convex search (ACS) algorithm to obtain the optimized solution with low computational complexity. Numerical simulation results show that the proposed method can adjust the offloading strategy properly according to task preference, and obtain a lower total cost compared with the baseline algorithms.**

***Index Terms*—Internet of vehicles, mobile edge computing, resource allocation.**

I. Introduction

With the rapid growth of the computation-intensive and latency-sensitive mobile applications in Internet of vehicles (IoV), the requirements for IoV change from data sharing into data transmitting and processing, which leads to the challenges of the computing and communication ability of IoV [1], [2]. mobile edge computing (MEC) has become a promising solution for providing sufficient computing resources for intelligent vehicles and meeting the low latency requirement for vehicle applications [3]–[5].

Through the support of MEC networks, intelligent vehicles can offload their large latency-sensitive computing service tasks (L2SC) to MEC servers to reduce task execute latency and vehicle energy consumption [6]–[8]. Although MEC networks can improve computational performance, a roadside MEC server cannot meet the growing and dynamic offloading requirements of vehicle terminals (VTs) with high mobility in its coverage. Hence, edge collaboration should be introduced into MEC networks to solve this problem.

The existing researches on edge collaboration in IoV networks mainly focused on three collaboration methods: cloud and MEC servers [9]–[13], vehicles [14]–[19], and multi MEC servers. Cloud computing can enrich the computing resources in MEC-enabled IoV networks. However, it can not meet the low latency requirement. Computing resources such as vehicles can utilize idle resources, but they change frequently, which is lack stability. Therefore, the collaboration between multi MEC servers can enrich edge resources and has better stability than vehicle resources, becomes a feasible solution to meet the growing and dynamice service requirements of IoV networks. The authors of [20] proposed a network that gave users three offloading options to reduce the computation workload of the MEC system. In [21], the authors satisfied the offloading requirements of the vehicle by purchasing computing resources from an alternate MEC server. The work by Xiao *et al.* [22] provided MEC cooperation based on traffic heat awareness, which can purchase computing resources from low-heat MEC servers to help high-heat MEC servers. The above researches mainly focused on workload balancing between multi MEC servers, but they didn't considered task splitting to further optimize the offloading strategy.

Considering the wide application of edge collaboration in the IoV-MEC network, splitting tasks and processing them by multiple edge devices will further improve the efficiency of task processing. However, where the task is split and how each subtask is offloaded and processed is still an open issue. Existing researches can be categorized into four types: serial offloading serial execution, serial offloading parallel execution, parallel offloading serial execution, and parallel offloading parallel execution. A further offloading strategy was proposed in [23] to jointly optimize latency and energy consumption by serial offloading serial execution. The authors of [24] provided a parallel offloading serial execution strategy to minimize the weighted sum of latency energy consumption. To minimize transmission energy consumption, a parallel offloading parallel execution strategy was provided in [25]. The authors of [26] combined cloud computing with MEC to achieve workload balancing through parallel offloading and parallel execution. The work by Chai *et al.* [27] proposed a parallel offloading parallel execution scheme, and categorized the tasks into different priorities to minimize the maximum task completion

Manuscript received September 29, 2021; revised November 20, 2022; approved for publication by Jalel Ben-Othman, Division 2 Editor, January 4 2023.

The work of this paper is funded by 2020 Industrial Technology Foundation Public Service Platform Project (2020-0105-2-1).

Y. Chen is with the School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China and Shunde Innovation School, University of Science and Technology Beijing, Guangdong 528399, China, email: chenyy@ustb.edu.cn

T. Deng, G. Chen, M. Yang, and L. Du are with School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China, email: dengtaoyu@126.com, gchen_ustb@foxmail.com, ustb_yangmj@163.com, lpdu200@163.com.

Y. Chen is the corresponding author.

Digital Object Identifier: 10.23919/JCN.2023.000004

1229-2370/23/$10.00 © 2023 KICS

time. The authors of [28] focused on reliability, where task offloading and splitting were controlled through the SDN network. And tasks were serially offloaded to access RSU and split to multi-device for parallel execution. When VT moves to the coverage of a new RSU, the services of MEC have to migrate to a new server, this process is called handover. However, this paper didn't consider the handover caused by vehicle mobility, the results of each subtask were still returned to the access RSU to be sent back to the vehicle.

Considering the high mobility and frequent handover in IoV networks. Some works have studied mobility prediction to help avoid handover latency and reduce retransmitting energy consumption, especially in MEC-enabled IoV networks [29]–[31]. There were also some studies focused on information source selection algorithm to enhance transmission efficiency and reliability in the dynamically changing network topology [32], [33]. Moreover, in IoV networks, different applications have different performance requirements. Hence, flexible offloading strategies are essential to satisfy the requirements of various applications while saving energy in IoV networks. The most above-mentioned researches only focused on latency or energy consumption. The authors of [34] proposed the weighted sum of latency and energy consumption in a cellular network. The tradeoff between latency and energy consumption was introduced in IoV networks by the authors of [20], [35]. However, they didn't consider edge cooperation.

Motivated by the above analysis, handover is frequent in IoV networks and will cause the decrease of offloading efficiency, but little literature studied the offloading strategy during handover. In order to meet the VT's offloading requirement with high mobility, we propose an offloading strategy using the RSUs located on the VT's pathway, in which RSUs on VT's pathway and other nearby RSUs are participated in computing during handover. This requires task splitting and transmitting subtasks to different RSUs to execute. Because a lot of uplink wireless resources will be occupied by parallel offloading, and tasks such as image recognition in IoV do not need to be executed sequentially [36], we propose a serial offloading parallel executing strategy to enhance task execution efficiency. Furthermore, as flexible offloading strategy is required to meet the various requirement of IoV applications, we propose to use multi MEC cooperation and adjust the weight sum factor based on task preference. Therefore, in this article, to achieve the various requirements of VT's applications with high mobility, we propose a serial offloading parallel execution strategy during the handover with latency energy tradeoff in MEC based IoV network. Specifically, we propose a MEC-based serial offloading parallel execution strategy to enhance L2SC offloading service experience for VT, which can utilize the handover time for edge execution. To reach the tradeoff between user experience and energy efficiency, we jointly optimize energy consumption and latency. The main contributions of this paper are shown as follows.

- We propose a novel cooperative offloading MEC framework in IoV networks, in which the task can be serial transmitted to RSU and parallel executed in several cooperate MEC servers, which can take full advantage of roadside computing resources and utilize the handover to satisfy the VT's low latency requirements in IoV networks.
- We introduce latency and energy consumption tradeoff into the proposed framework, and we use the weighted sum of latency and energy consumption as the cost. This can flexibly adjust the weighting factor for different IoV tasks, to meet the various requirements of IoV applications while saving energy.
- Because of the non-smoothness and non-convexity of the formulated problem, we provide an alternate convex search (ACS)-based algorithm to divide the original problem into two convex subproblems and obtain the suboptimal solution by iteratively solving the two subproblems.

The rest of this paper is organized as follows. The system model is presented in Section II. The problem formulation and algorithm design are given in Section III and IV, respectively. The proposed method is verified by simulation results in Section V. Conclusions are drawn in Section VI.

II. System Model

An MEC-enabled IoV network is considered as illustrated in Fig. 1. There are one vehicle and several roadside units (RSU) equipped with MEC servers, which are called edge nodes. Edge nodes can work cooperatively to provide task offloading service to the vehicle. All edge nodes are connected with one high-speed optical fiber, and the transmission rate is equal between any two edge nodes. The vehicle has one task to offload to MEC servers when it is about to move into the coverage of a new RSU. In this paper, we aim to verify the performance of the proposed cooperative offloading MEC framework. Therefore, we only consider a one-car scenario. The multi-car scenarios will be studied in our future works.

To make the problem simple, we only consider the single task scenario in this article, and the multi-tasks scenario will be studied in our future works. We assume that the channel status remains unchanged while the task is uploading, for the reason that the coherence time is close to the uplink transmission time. Because of the high speed of vehicles, a vehicle may experience cell switching while offloading a L2SC task. A RSU and its associated MEC server constitute an edge node [37]. We assume that the MEC server will execute the input task as soon as it receives all the data of the task. We also assume that B_f represents the edge node who covered the VT when the task was produced, B_l represents the edge node who covered the VT when task offloading is over, and for other edge nodes who participated in the collaboration, we call them *supporter*. The task size (bits), the computation workload (cycles/bit) [2], and the latency constrain of the task are represented as C, α, $T_{\max}$, respectively. We also assume that the task is bit independent [24] and can be divided into several parts of arbitrary size after transmitted to B_f, one of which is executed at B_f and others are further offloaded to any other edge nodes. Because the task is bit independent, the subtasks do not need to be executed in a certain order. We assume that each edge node begins to execute the subtask at the time they receive it, which we called parallel execute. The results of each subtask will be transmit to B_l at the time it

finish execution and combined at B_l, finally B_l transmits the final result to VT.

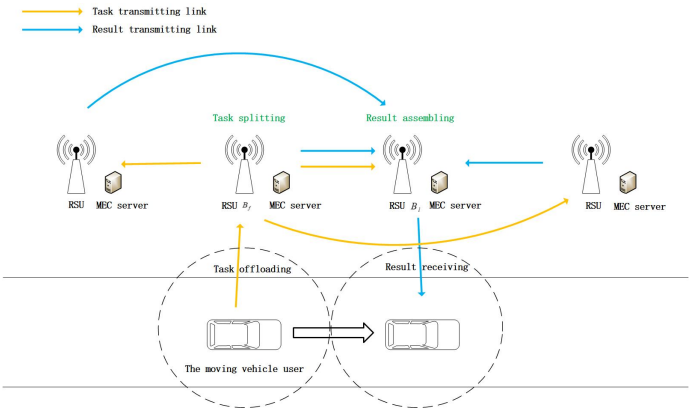

Fig. 1. System Model of multiple MEC-enabled IoV networks.

Let P denotes the transmit power from the VT to the edge node B_f. The uplink transmit time between the VT and edge node B_f can be given by

$$T_s = \frac{C}{R_s} = \frac{C}{Wlog_2(1+\frac{Ph^2}{N_0})}, \tag{1}$$

where $R_s = Wlog_2(1+Ph^2/N_0)$ is the uplink transmit rate, W, h, and N_0 represent the uplink bandwidth, uplink channel fading factor, and Gaussian noise power.

A. Task Splitting Model

To reduce the latency and save more energy, VT offloads the whole task to the edge. After being received by B_f , the task will be divided arbitrarily into K subtasks (K is the number of edge nodes that help execute the task), and transmit to other edge nodes through the high-speed optical fiber to execute separately. Let $\eta_m, m \in M$ with $0 < \eta_m < 1$ be the proportion of the subtask executed by edge node $B_m, m \in M$ to the original task size, and M is the set of all edge nodes. The data size and computation complexity that offloaded to edge node m are denoted by $\eta_m C$ and $\alpha\eta_m C$. Then, the sum of all fractions of the task should be equal to 1

$$\sum_{m\in M} \eta_m = 1. \tag{2}$$

B. Latency and Energy Consumption Model

1) Latency: The latency caused by task offloading can be separated into **chain latency** and **edge latency**. T_{chain} is the chain latency that represents the transmitting latency between VT and edge nodes. Because the MEC computing latency and the transmitting latency between edge nodes are both related to the number of MEC servers, we define the sum of them as edge latency, which is symbolized as T_{edge}.

Therefore, the total latency can be expressed as

$$T_{\text{total}} = T_{\text{chain}} + T_{\text{edge}}. \tag{3}$$

Chain latency includes uplink transmit latency and downlink transmit latency.

$$T_{\text{chain}} = T_s + T_x. \tag{4}$$

According to Shannon equation, the uplink transmit latency can be expressed as

$$T_s = \frac{C}{R_s} = \frac{C}{Wlog_2(1+\frac{Ph^2}{N_0})}, \tag{5}$$

where C represents the task data size, and uplink transmit speed is denoted by $R_s = Wlog_2(1+Ph^2/N_0)$. P, W, N_0, and h represent the vehicle's transmit power, uplink channel bandwidth, Gaussian noise power, and channel gain. Similarly, downlink transmitting latency can be expressed as

$$T_x = \frac{\gamma C}{R_x} = \frac{\gamma C}{Wlog_2(1+\frac{P_{EN}h^2}{N_0})}, \tag{6}$$

where P_{EN} represents the edge node's transmit power. Because the result data is much less than the task data, and the edge node's transmit power is much larger than the vehicle's transmit power, the downlink transmitting latency will be much smaller than the uplink transmitting latency. Hence, the downlink transmitting latency can be ignored.

The edge latency includes the time of subtask execution in MEC servers and the time of subtask transmission between the MEC servers. In the considering scenario, the edge node B_f receives the original task from the VT, divides the original task into m subtasks, then B_f executes the subtask that it undertakes and sends mth subtask to $B_m, m \in M/\{f\}$ simultaneously. Finally, B_f sends the result of its subtask to B_l. The edge node $B_i, i \in I$ receives the subtask sent by B_f. Then $B_i, i \in I$ executes the subtask that it undertakes. Finally, $B_i, i \in I$ sends the result of its subtask to B_l. B_l receives the subtask sent by B_f and the results of the tasks that $B_m, m \in M/\{l\}$ undertakes. Then B_l executes its subtask. Finally, B_l sends the final result of the original task to the VT. Because we consider parallel execution of the task, the edge latency is expressed as follows,

$$\begin{aligned} T_{\text{edge}} = max\Bigg\{ & \frac{\alpha\eta_f C}{f} + \frac{\gamma\eta_f C}{\beta}; \\ & \frac{\eta_i C}{\beta} + \frac{\alpha\eta_i C}{f} + \frac{\gamma\eta_i C}{\beta}; \frac{\alpha\eta_l C}{f} + \frac{\eta_l C}{\beta} \Bigg\} \qquad i \in I, \end{aligned} \tag{7}$$

where I is the set of all supporters, and η_f, η_i, and η_l denote the proportion of the subtask executed by edge nodes to the original task size. The sum of all fractions of the task should be equal to 1, i.e., $\eta_f + \eta_l + \sum_{i\in I}\eta_i = \sum_{m\in M}\eta_m = 1$. β, γ, and f (note to distinguish this f with the f in the footmark of B_f) represent the transmit speed on the high-speed optical fiber(bit/s) [38], the proportion between task result and itself [39], and the computation ability of MECs (rad/s) respectively. The first term $\alpha\eta_f C/f + \gamma\eta_f C/\beta$ in the $max\{\cdot\}$ of (7) is the edge latency caused by executing the subtask in the B_f. $\alpha\eta_f C/f$ is the subtask execution time, while $\gamma\eta_f C/\beta$ is execution result transmission time between the B_f and B_l. The second term $\eta_i C/\beta + \alpha\eta_i C/f + \gamma\eta_i C/\beta, i \in I$ in the $max\{\cdot\}$ of (7) is the edge latency caused by executing the subtask in the $B_i, i \in I$. $\eta_i C/\beta$ is the time of the

unexecuted subtask transferred from B_f to B_i. $\alpha\eta_i C/f$ is the subtask execution time. And $\gamma\eta_i C/\beta$ is the execution result transmission time between the B_i and B_l. The third term $\alpha\eta_l C/f + \eta_l C/\beta$ in the $max\{\cdot\}$ of (7) is the edge latency caused by executing the subtask in the B_l. $\eta_l C/\beta$ is the time of the unexecuted subtask transferred from B_f to B_l, while $\alpha\eta_l C/f$ is the subtask execution time.

2) Energy consumption: The energy consumption of the vehicle caused by offloading is mainly from two aspects, the uploading task energy consumption and the receiving result energy consumption. Then, the vehicle's total energy consumption can be expressed as

$$e_{\text{total}} = e_s + e_x, \tag{8}$$

where the upload task energy consumption e_s can be expressed as

$$e_s = PT_s. \tag{9}$$

Because the result data is much lesser than the task, and data receiving energy consumption is much lesser than data transmitting energy consumption, we ignore the vehicle's energy consumption e_x caused by receiving the result.

III. Problem Formulation

To reduce the waste of energy while ensuring QoE, we introduce a weighting factor δ and propose an optimization objective based on energy efficiency cost (EEC) [40]. The weighting factor δ ranges from [0,1], which value is determined by task preference. In other words, when the task is served for safety such as accident information analysis, the weighting factor δ will be set to a bigger value to satisfy the low latency demand of the task. Similarly, δ will be set to a smaller value when meeting entertainment task to save energy.

$$\varepsilon(\eta_m, P) = \delta T_{\text{total}} + (1-\delta)\mu e_{\text{total}}, \tag{10}$$

where $\varepsilon(\eta_m, P)$ represents the EEC, and weight balancing factor is denoted by μ. The energy efficiency cost $\varepsilon(\eta_m, P)$ is a unitless parameter, it reflects the latency and energy consumption tradeoff performance of the offloading strategy. To make delay and energy consumption have the same influence on EEC, we introduce a weight balancing factor μ. μ is defined as the ratio of the expected delay with $\delta = 1$ to the expected energy consumption with $\delta = 0$ [34], which keeps latency and energy consumption in the same order of magnitude. Therefore, when the value of δ changes, only the weight of delay and energy consumption will be changed, and the value of EEC will not be affected. Hence, tasks with different requirements can still be compared through EEC. When $\delta = 1$, $\varepsilon(\eta_m, P)$ only considers latency cost, and the optimization problem degenerates to a latency optimization problem. On the contrary, when $\delta = 0$, $\varepsilon(\eta_m, P)$ only relates to energy consumption cost, and the optimization problem degenerates to an energy consumption optimization problem.

As a result, the problem of minimizing EEC can be formulated as

$$\begin{aligned} \min_{\eta_m, P} \quad & \varepsilon(\eta_m, P) \quad m \in M \\ s.t. \quad & C1: \sum_{m\in M} \eta_m = 1 \\ & C2: 0 \le \eta_m \le 1 \\ & C3: 0 \le \delta \le 1 \\ & C4: T_{\text{total}} \le T_{\max} \\ & C5: 0 \le P \le P_{\max}, \end{aligned} \tag{11}$$

where C1 represents that the sum of all fractions of the VT's task should equal to 1, C2 and C3 represent the domains for η_m and δ ,C4 corresponds to the stringent latency constraint for the VT's task, and C5 corresponds to the stringent transmit power constraint for the VT.

The difficulties of solving this problem mainly lie in the non-convexity and non-smoothness of the objective function, and it is impractical to obtain the globally optimal solution by resorting to conventional optimization techniques. Hence, we apply an alternate convex search based algorithm to converge a sub-optimal solution to the problem. To be more specific, we first introduce an auxiliary variable T_n to convert the original problem to a smooth optimization problem, and then we transform the initial problem into two convex subproblems with low computation complexity by fixing some variable blocks, after that we use an ACS algorithm to obtain the optimized solution by iteratively solving two subproblems.

IV. Algorithm

According to the alternate convex search based algorithm, we first introduce an auxiliary variable T_n, which satisfies $T_{dege} + T_s \le T_n + T_s \le T_{\max}$ to convert the original problem (11) to a smooth optimization problem.

$$\begin{aligned} \min_{\eta_m, T_n, P} \quad & \varepsilon'(\eta_m, T_n, P) \quad m \in M \\ s.t. \quad & C1: \sum_{m\in M} \eta_m = 1 \\ & C2: 0 \le \eta_m \le 1, m \in M \\ & C3: 0 \le \delta \le 1 \\ & C5: 0 \le P \le P_{\max} \\ & C7: T_n + T_s \le T_{\max} \\ & C8: \frac{\gamma\eta_f C}{\beta} + \frac{\alpha\eta_f C}{f} \le T_n \\ & C9: \frac{\eta_i C}{\beta} + \frac{\alpha\eta_i C}{f} + \frac{\gamma\eta_i C}{\beta} \le T_n \quad i \in \boldsymbol{I} \\ & C10: \frac{\alpha\eta_l C}{f} + \frac{\eta_l C}{\beta} \le T_n, \end{aligned} \tag{12}$$

where

$$\varepsilon'(\eta_m, T_n, P) = \delta(T_n + T_s) + (1-\delta)\mu P T_s. \tag{13}$$

In the converted problem (12), we add three complementary constraints C8, C9, and C10 based on $T_n + T_s \le T_{\max}$, and replace T_{total} with $T_n + T_s$ in the optimization objective and

Algorithm 1 The alternate convex search based algorithm

Require: Network parameters δ, α, γ, etc; convergence tolerance ζ, iteration index $\tau = 1$

Ensure: η_m, P

1: Initialize starting variables η_m^0, T_n^0, P^0 and a^0
2: **repeat**
3: Update η_m^τ and T_n^τ according to linear problem (14)
4: Update a^τ by solving convex problem (18)
5: Obtain P according to problem(17)
6: **until** $\varepsilon^{'}(\eta_m^\tau, T_n^\tau, P^\tau) - \varepsilon^{'}(\eta_m^{\tau-1}, T_n^{\tau-1}, P^{\tau-1}) \leq \zeta$
7: **return** η_m, P

constraint C4 (C4 was changed to C7 after this operation). The transformed problem (12) is a smooth optimize problem which has no max function in its objective function.

However, the converted problem (12) is still non-convex and hard to solve. We need to convert the smooth optimize problem to a smooth biconvex problem by fixing some variable blocks.

A. The ACS Based Algorithm

1) Update η_m, T_n: Let τ denotes the τth iteration. $P^{\tau-1}$ is determined in the $\tau-1$th iteration. Substituting them into (12), we can obtain a linear problem with respect to η_m and T_n, which is given by

$$\begin{aligned} \underset{\eta_m, T_n}{min} \quad & \delta T_n \quad m \in M \\ s.t. \quad & C1: \sum_{m \in M} \eta_m = 1 \\ & C2: 0 \leq \eta_m \leq 1 \\ & C3: 0 \leq \delta \leq 1 \\ & C7: T_n + T_s^{\tau-1} \leq T_{\max} \\ & C8: \frac{\gamma\eta_f C}{\beta} + \frac{\alpha\eta_f C}{f} \leq T_n \\ & C9: \frac{\eta_i C}{\beta} + \frac{\alpha\eta_i C}{f} + \frac{\gamma\eta_i C}{\beta} \leq T_n \quad i \in I \\ & C10: \frac{\alpha\eta_l C}{f} + \frac{\eta_l C}{\beta} \leq T_n. \end{aligned} \tag{14}$$

Problem (14) is a classic optimization problem and can be solved by many efficient methods with low computation complexity, e.g., the simplex method [41]. But in order to better express its mathematical characteristics, we give the analytical solution of this problem as follows:

$$\begin{aligned} T_n =& max \left\{ \frac{\alpha\eta_f C}{f} + \frac{\gamma\eta_f C}{\beta}; \right. \\ & \left. \frac{\eta_i C}{\beta} + \frac{\alpha\eta_i C}{f} + \frac{\gamma\eta_i C}{\beta}; \frac{\alpha\eta_l C}{f} + \frac{\eta_l C}{\beta} \right\} \qquad i \in I \\ \eta_l =& \frac{1}{\frac{f+\alpha\beta}{\gamma f+\alpha\beta} + J\frac{f+\alpha\beta}{(1+\gamma)f+\alpha\beta} + 1}, \end{aligned} \tag{15}$$

where $\eta_f = \frac{f+\alpha\beta}{\gamma f+\alpha\beta}\eta_l$, $\eta_i = \frac{f+\alpha\beta}{(1+\gamma)f+\alpha\beta}\eta_l$, and J represents the number of supporters. The proof of this solution is shown in Appendix A.

2) Update P: We can obtain η_m^τ , T_n^τ by solving the linear programming problem (14). Substituting them into (12), we can obtain an optimization problem with respect to P

$$\begin{aligned} \underset{P}{min} \quad & \delta(T_n^\tau + T_s) + (1-\delta)\mu P T_s \\ & = \delta(T_n^\tau + \frac{C}{Wlog_2(1+\frac{Ph^2}{N_0})}) \\ & \quad + (1-\delta)\mu P \frac{C}{Wlog_2(1+\frac{Ph^2}{N_0})} \\ s.t. \quad & C3: 0 \leq \delta \leq 1 \\ & C5: 0 \leq P \leq P_{\max} \\ & C7: T_n^\tau + T_s \leq T_{\max}. \end{aligned} \tag{16}$$

The form of problem (16) is complicated so that we have to use variable substitution before we confirm the convexity of (16). Hence, we introduce a variable

$$a = \frac{1}{R_s} = \frac{1}{Wlog_2(1+\frac{Ph^2}{N_0})}, \tag{17}$$

and problem (16) can be transformed to a convex optimization problem with respect to a

$$\begin{aligned} \underset{a}{min} \quad & \delta(T_n^\tau + Ca) + (1-\delta)\mu\frac{N_0}{h^2}Ca(2^{\frac{1}{aW}} - 1) \\ s.t. \quad & C3: 0 \leq \delta \leq 1 \\ & C11: \frac{1}{Wlog_2(1+\frac{P_{\max}h^2}{N_0})} \leq a \\ & C12: Ca + T_n^\tau \leq T_{\max}, \end{aligned} \tag{18}$$

where P is replaced by $\frac{N_0}{h^2}(2^{\frac{1}{aW}} - 1)$ in the optimization objective and constraints (C5, C7 were changed to C11, C12 after this operation).

Furthermore, we can derive that when $f^{'}(a) = 0$ problem (18) obtains the optimal solution:

$$2^{\frac{1}{aW}}(1 - \frac{ln2}{aW}) = \frac{\delta h^2}{(\delta-1)N_0} + 1. \tag{19}$$

The proof of the convexity and solution of problem (18) are shown in Appendix B.

In summary, because (18) is equivalent to (12) if $\eta_m = \eta_m^\tau$ and $T_n = T_n^\tau$ are the optimal solutions to (14). The original problem can converge to a sub-optimal solution by iterating between (14) and (18) after converted into a smooth optimization problem (12). The process of the ACS based algorithm is shown in Algorithm 1.

The Algorithm 1 is still useful when $\delta = 1$, and when $\delta = 0$ the original problem degenerates to a convex optimization problem and can be solved by many efficient methods with low computation complexity. The proof is shown in Section IV.B and Section IV.C.

B. Special Case: $\delta = 1$

The smooth optimization problem (12) converts to a new optimization problem, and the proposed algorithm is still suitable.

$$\begin{aligned} \underset{\eta_m, T_n, P}{min} \quad & T_n + T_s \quad m \in M \\ s.t. \quad & C1: \sum_{m \in M} \eta_m = 1 \\ & C2: 0 \le \eta_m \le 1 \\ & C5: 0 \le P \le P_{\max} \\ & C7: T_n + T_s \le T_{\max} \\ & C8: \frac{\gamma \eta_f C}{\beta} + \frac{\alpha \eta_f C}{f} \le T_n \\ & C9: \frac{\eta_i C}{\beta} + \frac{\alpha \eta_i C}{f} + \frac{\gamma \eta_i C}{\beta} \le T_n \quad i \in \boldsymbol{I} \\ & C10: \frac{\alpha \eta_l C}{f} + \frac{\eta_l C}{\beta} \le T_n. \end{aligned} \tag{20}$$

1) Update η_m, T_n: We can obtain a linear problem.

$$\begin{aligned} \underset{\eta_m, T_n}{min} \quad & T_n \quad m \in M \\ s.t. \quad & C1: \sum_{m \in M} \eta_m = 1 \\ & C2: 0 \le \eta_m \le 1 \\ & C7: T_n + T_s^{\tau} \le T_{\max} \\ & C8: \frac{\gamma \eta_f C}{\beta} + \frac{\alpha \eta_f C}{f} \le T_n \\ & C9: \frac{\eta_i C}{\beta} + \frac{\alpha \eta_i C}{f} + \frac{\gamma \eta_i C}{\beta} \le T_n \quad i \in \boldsymbol{I} \\ & C10: \frac{\alpha \eta_l C}{f} + \frac{\eta_l C}{\beta} \le T_n. \end{aligned} \tag{21}$$

2) Update P: By introducing an auxiliary variable as we did in Section IV, we can obtain a convex optimization problem.

$$\begin{aligned} \underset{a}{min} \quad & \delta(T_n^{\tau} + Ca) \\ s.t. \quad & C3: 0 \le \delta \le 1 \\ & C11: \frac{1}{W log_2(1 + \frac{P_{\max} h^2}{N_0})} \le a \\ & C12: Ca + T_n^{\tau} \le T_{\max}. \end{aligned} \tag{22}$$

The original problem can converge to a sub-optimal solution by iterating between the above two convex problems after converted into a smooth optimization problem (12). The proof is finished.

C. Special Case: $\delta = 0$

The original problem (11) convert to a new convex optimization problem.

$$\begin{aligned} \underset{\eta_m, P}{min} \quad & \mu P T_s \quad m \in M \\ s.t. \quad & C1: \sum_{m \in M} \eta_m = 1 \\ & C2: 0 \le \eta_m \le 1 \\ & C5: 0 \le P \le P_{\max} \\ & C13: \frac{\gamma \eta_f C}{\beta} + \frac{\alpha \eta_f C}{f} + T_s \le T_{\max} \\ & C14: \frac{\eta_i C}{\beta} + \frac{\alpha \eta_i C}{f} + \frac{\gamma \eta_i C}{\beta} + T_s \le T_{\max} \quad i \in \boldsymbol{I} \\ & C15: \frac{\alpha \eta_l C}{f} + \frac{\eta_l C}{\beta} + T_s \le T_{\max}. \end{aligned} \tag{23}$$

By introducing an auxiliary variable as we did in Section IV, the above problem can be transformed to an optimization problem with respect to a and η_m.

$$\begin{aligned} \underset{a, \eta_m}{min} \quad & \mu \frac{N_0}{h^2} Ca(2^{\frac{1}{aW}} - 1) \\ s.t. \quad & C1: \sum_{m \in M} \eta_m = 1 \\ & C2: 0 \le \eta_m \le 1 \\ & C11: \frac{1}{W log_2(1 + \frac{P_{\max} h^2}{N_0})} \le a \\ & C13: \frac{\gamma \eta_f C}{\beta} + \frac{\alpha \eta_f C}{f} + Ca \le T_{\max} \\ & C14: \frac{\eta_i C}{\beta} + \frac{\alpha \eta_i C}{f} + \frac{\gamma \eta_i C}{\beta} + Ca \le T_{\max} \quad i \in \boldsymbol{I} \\ & C15: \frac{\alpha \eta_l C}{f} + \frac{\eta_l C}{\beta} + Ca \le T_{\max}. \end{aligned} \tag{24}$$

Problem (24) is a convex problem, which can be proved by Appendix B. The proof is finished.

D. Complexity Analysis of Algorithm 1

Because Algorithm 1 solves the original problem by iterate solving problem (14) and problem (16), the computation complexity of Algorithm 1 is equal to the sum of the computation complexity of these two problems. Problem (14) is a linear problem, which has the complexity of $O(a^2 b)$, where a is the number of variables and b is the number of constraints. The interior-point method is used for solving the convex problem (16), and it has the complexity of $O(n^{3.5} log(1/\epsilon))$, where n is the variable dimension, and ϵ is the target precision [42]. Therefore, the complexity of Algorithm 1 is $O(28 + log(1/\epsilon))$ where ϵ is the target precision.

V. Simulation Results

In this section, we provide simulation results to evaluate the performance of the proposed partial offloading strategy by comparing it to three baseline strategies. We assume that there

TABLE I
KEY PERFORMANCE INDICATORS.

Parameters	Values
δ weighting factor	0–1
α computation workload	40 cycles/bit
γ proportion between task result and itself	0.2
β transmit speed on the high-speed optical fiber	10^{10} bit/s
C task data size	1–10 Mbits
Transmission frequency	5.9GHz
f computation ability of MECs	8×10^9 cycles/s
h channel gain	Rayleigh fading channel
N_0 power spectral density	3×10^-13 W
$T_{\max}$ vehicle's task latency upper bound	30 ms
$P_{\max}$vehicle's transmit power upper bound	0.2 W

is a vehicle traveling down a one-way straight highway at the speed of about 100 km/h, and the coverage of RSU is 10 m. The transmission frequency between VT and edge nodes is 5.9 GHz. During the L2SC task offloading, the vehicle will experience handover, and receive the computing result after the handover. According to the existing researches, we express the existing offloading strategies into three baseline algorithms:

Baseline method 1 (no cooperation strategy): All fractions of the task are computed at the node B_f after the task is offloaded to it, then transmit the result to the node B_l, and then transmit the result to the VT. It represents the recent offloading strategies that didn't consider MEC cooperation.

Baseline method 2 (further offloading strategy): The task is first offloaded to the node B_f and transmit all fractions of the task to node B_l , then computing all the task at node B_l, and then transmit the result to the VT. It represents the recent offloading strategies that used MEC binary offloading or further offloading. This method didn't consider task splitting in MEC cooperation.

Baseline method 3 (serial cooperate strategy): The task is split at the VT and offloading to several edge nodes parallelly, then MEC servers execute each part of the task serially in a fixed order. Finally, each part of the task result transmits to node B_l and then transmit the result to the VT. It represents the recent offloading strategies that have considered MEC cooperation by the serial computing of MEC servers.

For the parallel cooperate offloading which we proposed, we also give the performance comparison between the schemes with different number of edge nodes.

The detailed simulation parameters are given in Table I.

Fig. 2 represents the latency performance in terms of the task data size. It shows the task data size that can be executed by different methods while meeting the 20 ms delay requirement of IoV application. We can also see that latency increases with the increase of task data size and the curve is linear. This is because task data size C is in direct proportion to T_{total}. As we deduced in part II, $T_{\text{total}} = T_{\text{edge}} + T_s$, and we can obtain from equation (5) and equation (7) that C is in direct proportion to T_{edge} and T_s. Hence, C is in direct proportion to T_{total}, which makes the curve linear. And proposed MEC cooperation strategy shows less latency than the three baseline strategies. Moreover,with the increase of the number of cooperation MECs, latency performance shows more superiority. This is because the proposed strategy splits

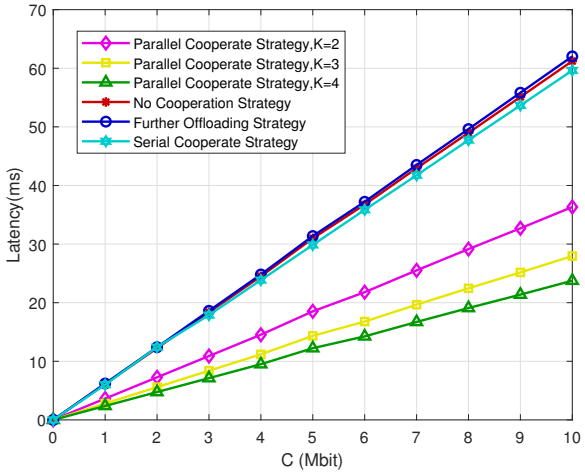

Fig. 2. Latency performance versus task data size.

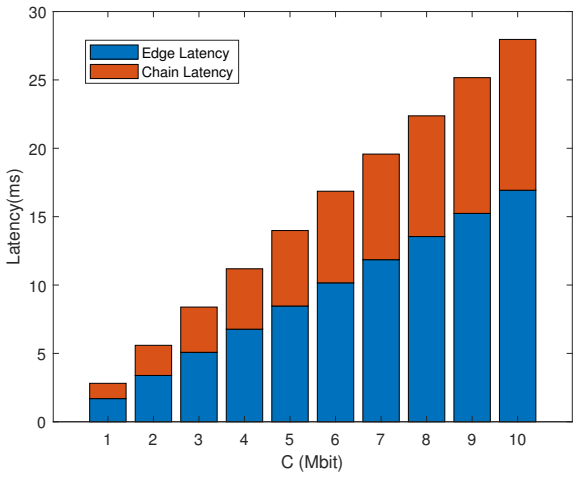

Fig. 3. Proportion of chain&edge latency in latency performance.

the tasks properly and uses the resources more rationally.

The latency performance in terms of the task data size is shown in Fig. 3, in which K is equal to 3. It indicates the proportion of chain latency and edge latency in Fig. 2. We can observe that edge latency occupies more proportion of the total latency, and the proportion between chain latency and edge latency remains unchanged with the increase of task data size. This is because the task preference index and the number of MEC servers remain unchanged, the proportion will change when these two parameters change.

Fig. 4 represents the total cost performance in terms of latency requirement. As shown in Fig. 4, the total cost first decreases and then remains unchanged as the latency requirement $T_{\max}$ increases. We can also see that, the proposed strategy can reach more rigorous latency requirements while consuming less total cost. It can be concluded from Fig. 4 that, the proposed MEC cooperation partial offloading strategy can satisfy the requirement of L2SC tasks while saving energy, which is more suitable for L2SC tasks.

The energy consumption performance in terms of latency requirement is illustrated in Fig. 5. We can observe that energy

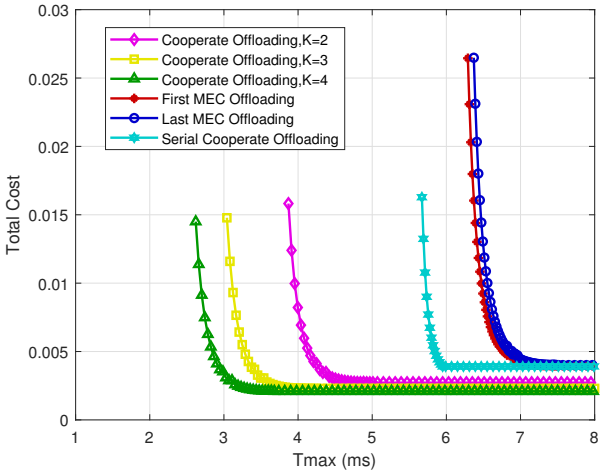

Fig. 4. Total cost performance versus latency requirement.

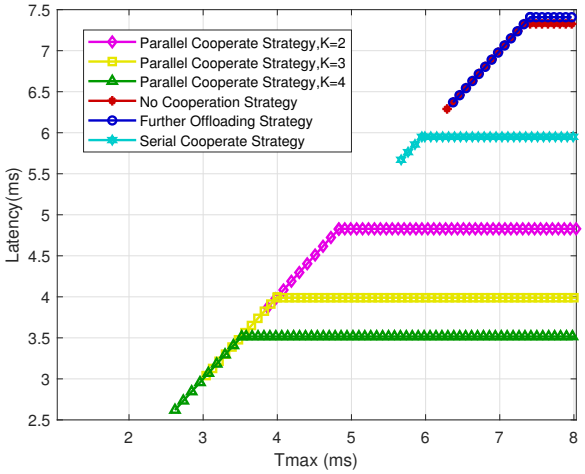

Fig. 6. Latency performance versus latency requirement.

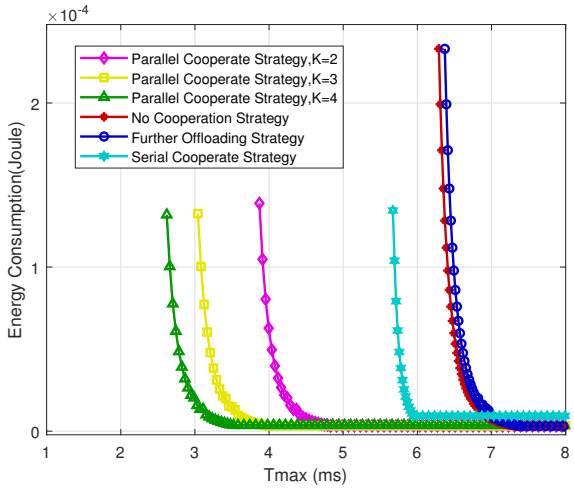

Fig. 5. Energy consumption performance versus latency requirement.

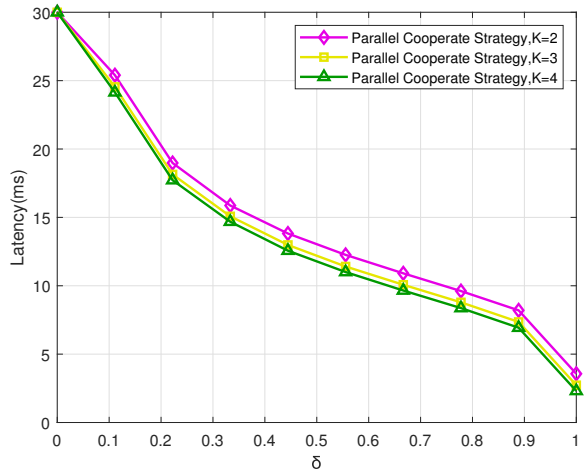

Fig. 7. Latency performance versus task preference index.

consumption first decreases and then remains unchanged as the latency requirement increases. We can also see that, comparing with the three baseline strategies, the proposed strategy can reach more rigorous latency requirements when the energy consumption is the same, and the proposed strategy shows more superiority when approaching the minimum latency requirement that the strategy can reach(more suitable for low latency scene). Besides, with the increase of the number of cooperation MECs, energy consumption performance shows more superiority.

Fig. 6 represents the latency performance in terms of latency requirement. It can be seen that latency first increases and then remains unchanged as the latency requirement $T_{\max}$ increases. We can also see that the proposed strategy shows less latency than the three baseline strategies and can satisfy more rigorous latency requirements, and with the increase of the number of cooperation MECs, latency performance shows more superiority.

The latency performance in terms of task preference is shown in Fig. 7. It shows that latency decreases with the increase of task preference index. This is because a large δ indicates the user's low latency tolerance, where we should enhance our latency performance to meet the user's requirement.Besides, with the increase of the number of cooperation MECs, latency performance shows more superiority.

The energy consumption performance in terms of task preference index is shown in Fig. 8. It indicates that energy consumption increases with the increase of task preference index. This is because a small δ indicates the user's low energy consumption tolerance, where we should enhance our energy consumption performance to meet the user's requirement.

Figs. 9 and 10 show the influence of different task preference indexes for the proposed method(where we set $K = 3$ as an example). We can observe from Fig. 9 that the proposed method can reach a better latency performance when choosing a higher task preference index. And with the increase of the task data size, latency performance shows more superiority. Fig. 10 shows the energy consumption performance in terms of task data size for different task preference indexes. It shows that the proposed method can reach a better energy consumption performance when choosing a lower task preference index, and with the increase of the task data size, energy

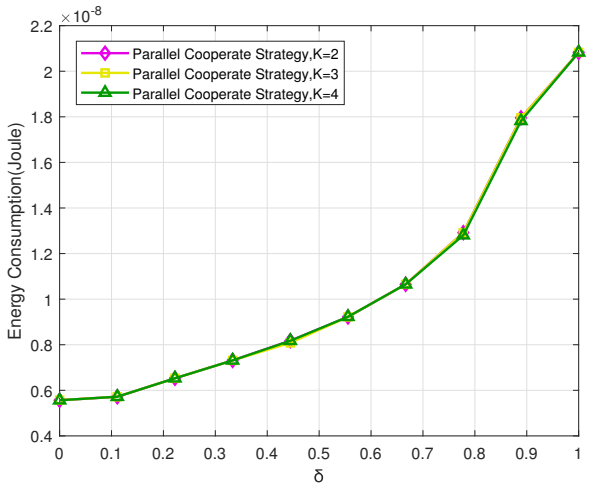

Fig. 8. Energy consumption performance versus task preference index.

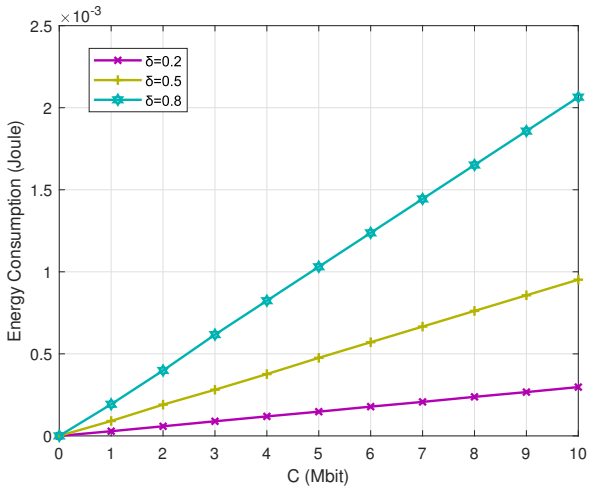

Fig. 10. Energy consumption performance versus task data size for different task preference indexes.

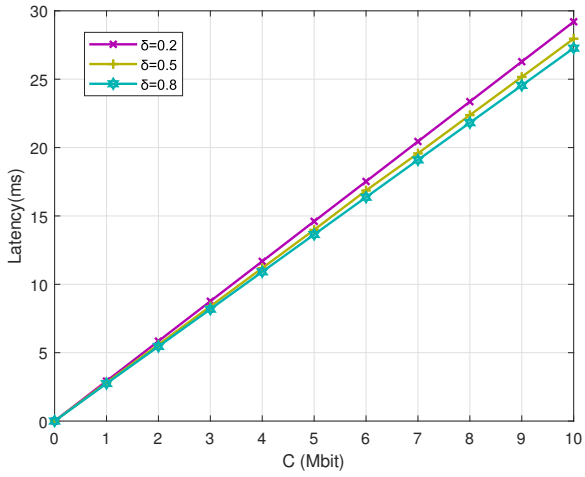

Fig. 9. Latency performance versus task data size for different task preference indexes.

consumption performance shows more superiority.

VI. Conclusion

In this paper, we investigate an edge collaborative task serial offloading parallel executing strategy in MEC-enabled IoV networks, which can split the task on the edge and use several MEC servers to paralleling execute each part of the task. And we formulate the problem as the minimization of the weighted sum of the energy consumption and the latency which is non-smooth and non-convex. An alternate convex search algorithm is provided to tackle the problem efficiently, which can converge to a sub-optimal solution. The numerical simulation results show that the proposed multi-MEC cooperating partial offloading strategy can take advantage of the roadside computing resources properly, and shows superiority in the weighted sum of latency and energy consumption. When latency requirement becomes more relaxed, the proposed strategy can further reduce the total cost. And for different tasks, the proposed strategy can also change dynamically to meet their needs by adjusting the task preference index. Moreover, the impacts of various parameters were revealed, which validate the feasibility of the proposed method in different situations. For future investigation, we plan to study the multi-task condition, and schedule the offloading sequence based on priority and task sequence. To solve the multi-task arriving problem, we will also study queuing issues.

Appendix A
Proof of the Analytical Solution of Problem (14)

It is obvious that the optimal solution of problem (14) is obtain at the minimum point of T_n , and we can derive from C8, C9, and C10 that

$$T_n = max\left\{\frac{\alpha\eta_f C}{f}+\frac{\gamma\eta_f C}{\beta};\ \frac{\eta_i C}{\beta}+\frac{\alpha\eta_i C}{f}+\frac{\gamma\eta_i C}{\beta};\frac{\alpha\eta_l C}{f}+\frac{\eta_l C}{\beta}\right\} \qquad i\in I, \tag{25}$$

to reach the minimum value of this max function, we need each part of the fuction to take the same value:

$$\begin{aligned}\frac{\alpha\eta_f C}{f}+\frac{\gamma\eta_f C}{\beta} &= \frac{\eta_i C}{\beta}+\frac{\alpha\eta_i C}{f}+\frac{\gamma\eta_i C}{\beta}\\ &= \frac{\alpha\eta_l C}{f}+\frac{\eta_l C}{\beta}.\end{aligned} \tag{26}$$

Based on function (21), we can obtain that

$$\begin{aligned}\eta_f &= \frac{f+\alpha\beta}{\gamma f+\alpha\beta}\eta_l\\ \eta_i &= \frac{f+\alpha\beta}{(1+\gamma)f+\alpha\beta}\eta_l.\end{aligned} \tag{27}$$

Finally, we can derive from C1, C2, and $\eta_f+\eta_l+\sum_{i\in I}\eta_i=1$ that

$$\eta_l = \frac{1}{\frac{f+\alpha\beta}{\gamma f+\alpha\beta}+J\frac{f+\alpha\beta}{(1+\gamma)f+\alpha\beta}+1}. \tag{28}$$

APPENDIX B
PROOF OF THE CONVEXITY AND ANALYTICAL SOLUTION OF PROBLEM (18)

Proof of the convexity: Let $f(a)$ represent the objective of (18) as

$$f(a) = \delta(T_n^\tau + Ca) + (1-\delta)\mu\frac{N_0}{h^2}Ca(2^{\frac{1}{aW}} - 1). \quad (29)$$

The second-order derivative of $f(a)$ in terms of a can be written as

$$f^{"}(a) = (1-\delta)\mu\frac{N_0}{h^2}C\frac{(ln2)^2}{a^3W^2}2^{\frac{1}{aW}}. \quad (30)$$

It is easy to obtain that the second-order derivative is always positive because $a > 0$, and it is also obvious that the constraints are convex. Hence, $f(a)$ is a convex function, and problem (18) is a convex optimization problem. The proof is finished.

Proof of the analytical solution: Because the second order derivative of $f(a)$ is always positive, the first order derivative is a monotonously increasing function with a. Moreover, the first order derivative satisfies $\lim_{a\to+\infty} f^{'}(a) = \delta c > 0$ and $\lim_{a\to 0} f^{'}(a) \to -\infty < 0$. Therefore, $f^{'}(a)$ has and only has one zero point, which is the unique minum point of $f(a)$. As a result, the minum value of $f(a)$ is obtained at its minum point where $f^{'}(a) = 0$. The proof is finished.

REFERENCES

[1] P. Papadimitratos, A. D. La Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *IEEE Commun. Mag.*, vol. 47, no. 11, pp. 84–95, 2009.

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

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

[4] S. Bitam, A. Mellouk, and S. Zeadally, "'VANET'-cloud: A generic cloud computing model for vehicular ad hoc networks," *IEEE Wireless Commun.*, vol. 22, no. 1, pp. 96–102, 2015.

[5] F. Spinelli and V. Mancuso, "Toward enabled industrial verticals in 5G: A survey on MEC-based approaches to provisioning and flexibility," *IEEE Commun. Surveys Tut.*, vol. 23, no. 1, pp. 596–630, 2021.

[6] Y. Wu and J. Zheng, "Modeling and analysis of the uplink local delay in MEC-based VANETs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 3538–3549, 2020.

[7] S. Xu *et al.*, "RJCC: Reinforcement-learning-based joint communicational-and-computational resource allocation mechanism for smart city IoT," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 8059–8076, 2020.

[8] D. Sabella *et al.*, "MEC-based infotainment services for smart roads in 5G environments," in *Proc. IEEE VTC*, 2020, pp. 1–6.

[9] X. Kong *et al.*, "Deep reinforcement learning-based energy-efficient edge computing for Internet of vehicles," *IEEE Trans. Ind. Informat.*, vol. 18, no. 9, pp. 6308–6316, 2022.

[10] R. W. L. Coutinho and A. Boukerche, "Modeling and analysis of a shared edge caching system for connected cars and industrial IoT-based applications," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2003–2012, 2020.

[11] S. M. A. Kazmi *et al.*, "Infotainment enabled smart cars: A joint communication, caching, and computation approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8408–8420, 2019.

[12] K. Sasaki, N. Suzuki, S. Makido, and A. Nakao, "Vehicle control system coordinated between cloud and mobile edge computing," in *Proc. IEEE SICE*, 2016, pp. 1122–1127.

[13] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, 2019.

[14] Y. Wang, X. Hu, L. Guo, and Z. Yao, "Research on V2I/V2V hybrid multi-hop edge computing offloading algorithm in IoV environment," in *Proc. IEEE ICITE*, 2020, pp. 336–340.

[15] C. Chen, Y. Zeng, H. Li, Y. Liu, and S. Wan, "A multi-hop task offloading decision model in MEC-enabled Internet of vehicles," *IEEE Internet Things J.*, p. 1, 2022.

[16] X. Wang, Z. Ning, S. Guo, and L. Wang, "Imitation learning enabled task scheduling for online vehicular edge computing," *IEEE Trans. Mobile Comput.*, vol. 21, no. 2, pp. 598–611, 2022.

[17] Y. Li *et al.*, "Joint offloading decision and resource allocation for vehicular fog-edge computing networks: A contract-stackelberg approach," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 15 969–15 982, 2022.

[18] S. Olariu, T. Hristov, and G. Yan, "The next paradigm shift: From vehicular networks to vehicular clouds," *Mobile ad hoc networking: Cutting edge directions*, pp. 645–700, 2013.

[19] D. Han, W. Chen, and Y. Fang, "A dynamic pricing strategy for vehicle assisted mobile edge computing systems," *IEEE Wireless Commun. Lett.*, vol. 8, no. 2, pp. 420–423, 2019.

[20] H. Wang, Z. Lin, K. Guo, and T. Lv, "Computation offloading based on game theory in MEC-assisted V2X networks," in *Proc. IEEE ICC Workshops*, 2021, pp. 1–6.

[21] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proc. IEEE ICC*, 2017, pp. 1–6.

[22] Z. Xiao *et al.*, "Vehicular task offloading via heat-aware MEC cooperation using game-theoretic method," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2038–2052, 2020.

[23] W. Fan, Y. Liu, B. Tang, F. Wu, and Z. Wang, "Computation offloading based on cooperations of mobile edge computing-enabled base stations," *IEEE Access*, vol. 6, pp. 22 622–22 633, 2018.

[24] M. Qin *et al.*, "Service-oriented energy-latency tradeoff for IoT task partial offloading in MEC-enhanced multi-RAT networks," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1896–1907, 2021.

[25] M. Zeng and V. Fodor, "Parallel processing at the edge in dense wireless networks," *IEEE Open J. Commun. Soc.*, vol. 3, pp. 1–14, 2022.

[26] W. Zhang, G. Zhang, and S. Mao, "Joint parallel offloading and load balancing for cooperative-MEC systems with delay constraints," *IEEE Trans. Veh. Technol.*, vol. 71, no. 4, pp. 4249–4263, 2022.

[27] R. Chai, M. Li, T. Yang, and Q. Chen, "Dynamic priority-based computation scheduling and offloading for interdependent tasks: Leveraging parallel transmission and execution," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10970–10985, 2021.

[28] X. Hou *et al.*, "Reliable computation offloading for edge-computing-enabled software-defined IoV," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7097–7111, 2020.

[29] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4568–4578, 2018.

[30] S. Zhou *et al.*, "Short-term traffic flow prediction of the smart city using 5G internet of vehicles based on edge computing," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–10, 2022.

[31] S. D. A. Shah, M. A. Gregory, S. Li, R. d. R. Fontes, and L. Hou, "SDN-based service mobility management in MEC-enabled 5G and beyond vehicular networks," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13425–13442, 2022.

[32] J. Wang *et al.*, "Vehicular sensing networks in a smart city: Principles, technologies and applications," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 122–132, 2018.

[33] J. Wang, C. Jiang, Z. Han, Y. Ren, and L. Hanzo, "Internet of vehicles: Sensing-aided transportation information collection and diffusion," *IEEE Trans. Veh. Technol.*, vol. 67, no. 5, pp. 3813–3825, 2018.

[34] J. Zhang *et al.*, "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2633–2645, 2018.

[35] H. Wang, Z. Lin, K. Guo, and T. Lv, "Energy and delay minimization based on game theory in MEC-assisted vehicular networks," in *Proc. IEEE ICC Workshops*, 2021, pp. 1–6.

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

[37] Y. Wu and J. Zheng, "Modeling and analysis of the downlink local delay in MEC-based VANETs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 6619–6630, 2020.
[38] C. Song *et al.*, "Hierarchical edge cloud enabling network slicing for 5G optical fronthaul," *J. Opt. Commun. Netw.*, vol. 11, no. 4, pp. B60–B70, 2019. [Online]. Available: https://opg.optica.org/jocn/abstract.cfm?URI=jocn-11-4-B60
[39] W. He *et al.*, "Latency minimization for full-duplex mobile-edge computing system," in *Proc. IEEE ICC*, 2019, pp. 1–6.
[40] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *Proc. IEEE INFOCOM*, 2016, pp. 1–9.
[41] V. Chvatal, *et al.*, *Linear programming*. Macmillan, 1983.
[42] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, maxcut and complex semidefinite programming," *Mathematical Programming*, vol. 149, no. 1, pp. 47–81, 2015.

Taoyu Deng received the B.S. degree from University of Science and Technology Beijing, in 2019. He is currently pursuing his M.S degree with the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. His current research interests are mobile edge computing and Internet of vehicles.

Yueyun Chen is a Professor of Information and Communication Engineering at University of Science and Technology Beijing. She graduated with a B.S. degree in Radio Technology from South China University of Technology, and M.S. and Ph.D. degrees in Communication and Information System from Beijing Jiaotong University, respectively. Her research interesting includes wireless mobile communication, AI in wireless communications, radio signal processing, massive MIMO, wireless networks, space information and communication systems, massive MIMO, MEC, etc.

Guang Chen received the B.S. degree from University of Science and Technology Beijing, in 2018. He is currently pursuing the Ph.D. degree with the School of Computer and Communication Engineering, University of Science and Technology Beijing, China. His research interests include mobile edge computing, game theory, convex optimization.

Meijie Yang received the B.S. degrees in University of Science and Technology Beijing, China, in 2016. She is currently pursuing the Ph.D. degree in University of Science and Technology Beijing, China. Her research interests are mostly focused on physical layer, advanced waveforms techniques and signal processing for 5G systems.

Liping Du Liping Du, received her B.Eng. and MA.Sc degree from Zhengzhou University, P.R. China, in 1998 and 2001, and her Ph.D from the Department of Electrical Engineering, Beijing Institute of Technology, P.R. China, in 2005. From 2005 to 2006, she worked as a Research Associate in the Department of Electrical Engineering, City University of Hongkong, Hongkong, under the supervision of IEEE Fellow Hong Yan. In 2006, she joined the Department of Communication, University of Science and Technology Beijing. From 2014 to 2015, she visited the CRES Lab of the University of California, Los Angeles. Now She is currently with the School of Computer & Communication Engineering, University of Science and Technology Beijing as an associate professor. Her research interest covers wireless communication, signal processing, cognitive radio.