Delay-Sensitive Slicing Resources Scheduling Based on Multi-MEC Collaboration in IoV

Yan Liang[1], Xin Chen[1(✉)], Shengcheng Ma[2], and Libo Jiao[1]

[1] School of Computer Science, Beijing Information Science and Technology University, Beijing 100101, China
{chenxin,jiaolibo}@bistu.edu.cn
[2] School of Computer Science and Engineering, Beihang University, Beijing 100191, China
mashengcheng@buaa.edu.cn

Abstract. The emerging Vehicle to Infrastructure (V2I) technology supports the service of task offloading under the Internet of vehicle (IoV), which improves the computational efficiency of the task. Facing tasks with different demands, network slicing technology builds a variety of logic private networks on a unified infrastructure, divides and allocates resources according to different user needs, which improves the vehicle transmission efficiency. Nevertheless, the diversity of demand resources and the randomness of tasks make the network scenario of IoV more complex. It is still a challenge to consider how to combine the network slicing technology to reduce the cost of offloading the computing task. In this paper, we study the scenario of autonomous vehicle offloading the computing task to Roadside Units (RSUs), and consider the multi-Mobile Edge Computing (multi-MEC) collaborative computing task to ensure that the task can be completed within tolerable delay. We consider the computing power and resource occupancy rate of MEC servers to ensure the user experience, and formulate a resource pricing scheme. Then, we propose a Performance-Price Ratio Task Scheduling (PPRTS) algorithm, which aims to complete the computing task within the maximum tolerable delay and reduce the cost of user. Simulation results show that the algorithm can effectively reduce the cost of the user.

Keywords: Internet of Vehicle (IoV) · Vehicle to Infrastructure (V2I) · Network slicing · Mobile Edge Computing (MEC) · Resource scheduling

1 Introduction

With the evolution of cellular network, the transmission rate of the network has been greatly improved, which provides technical support for the information interaction between vehicle information and external traffic elements [1]. V2I

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 407, pp. 50–64, 2021.
https://doi.org/10.1007/978-3-030-92638-0_4

communication is an important form of Vehicle to Everything (V2X), which is considered to improve the roadside safety and traffic system in the Intelligent Transportation System (ITS) [2]. When the network of RSU equipment is dense enough, it can act as the orientation point and provide the vehicle with relevant information about the dangerous situation on the road, such as traffic jam ahead, traffic accidents and other risks. In addition, the vehicles offload the computing task to the MEC server of RSU, which can process the data faster.

In V2I communication scenario, it mainly refers to the interactive communication between vehicles and RSUs. V2I can facilitate the vehicle to quickly obtain the surrounding facilities information and process the computing tasks. By connecting the RSU with the Internet, the RSU can be turned into a relay point, which reduces the dependence of communication between vehicles and base stations, vehicles and other vehicles in the Internet of vehicles network. Compared with vehicle to base station communication, V2I can reduce the transmission delay and the processing delay of computing tasks [3].

Moreover, RSU combined with Mobile Edge Computing (MEC) technology is used to improve the efficiency of task processing. As a new deployment scheme, MEC can reduce the core network load and data transmission delay by deploying small data centers or nodes with cache and computing capabilities at the edge of the network, which is closely connected with mobile devices and users [4]. The mobile terminal can judge whether it needs offloading service according to the delay tolerance of the task, processing capacity, energy consumption, and other factors. By employing offloading service, the computing-intensive and delay-sensitive tasks can be processed on MEC servers to meet the performance requirements of tasks [5].

However, Multiple users request and share resources from MEC server at the same time, which causes congestion and slow response speed. In addition, different users have different requirements for data capacity and computing resources, which leads to inefficient resource allocation. In order to solve this problem, network slicing technology is used to divide the data capacity and computing resource of MEC server into multiple slices. These slices are respectively allocated to multiple users, and they are isolated from each other [6]. Network slicing technology provides a powerful guarantee to solve the problem of different requirements in network capacity, delay, reliability, speed and other aspects in diversified application scenarios [7]. And it builds corresponding logical networks for different types of resource requirements on the same physical infrastructure, and ensures mutual isolation [8].

Resource pricing scheme can affect the cost and resource utilization strategy of users, which is an important part. In [9], Baek B. *et al.* considered three dynamic pricing mechanisms for edge computing resource allocation in the Internet of Things environment: bid-proportional allocation mechanism, uniform pricing mechanism, and fairness-seeking differentiated pricing mechanism. In [10], Cardellini V. *et al.* studied the resource pricing and Provisioning Strategies in cloud systems, and found that the dynamic pricing scheme for different customers can give network operator higher income. By exploring the relationship between resource efficiency and profit maximization, Wang G. *et al.* studied the dimension-

ing of network slicing with resource pricing strategy in [11], and improved the profit of network slicing in [12]. However, most of these pricing schemes aim at improving profits of operator, without considering resource utilization and user competition, so we adopt dynamic asymmetric pricing strategy to improve resource utilization.

At present, some authors consider some strategies and methods to complete computing tasks. In [13], Liu M. *et al.* considered the edge cloud with limited computing power, divided the spectrum resources and computing resources of the edge cloud MEC server, and sold them to multiple users. And the partial offloading strategy was adopted to divide the user's computing tasks into different parts for local computing and offloading at the same time. In [14], Liu Y. *et al.* introduced the non-orthogonal multiple access function, which ensured that multiple users can offload computing tasks to the same fog node, so that a fog node can serve multiple users. On this basis, the total system cost of energy and user delay are minimized. In [15], Wang C. *et al.* chose some user equipments to offload their computing tasks, while others execute their computing tasks locally. However, this does not take into account users's willingness to offload, so a reasonable pricing scheme should be made to guide users to choose offloading when the resource utilization rate is low, otherwise choose local computing.

Many authors think that cloud computing in core network may cause too much delay, and consider offloading computing tasks to RSU to provide low delay services. In [16], Huang C. *et al.* considered that the computing task is offloaded from the vehicle to the base station, which leads to the increase of data transmission time, so the computing task is offloaded to RSU. In [17], Chen C. *et al.* used the idle resources between vehicles for collaborative computing. Computing tasks can also be offloaded to a single RSU, but the collaborative computing of multiple RSUs is not considered, which causes great pressure on a single RSU.

Based on the above, in this paper we build a multi-RSU collaborative slicing resources scheduling model to solve the delay sensitive computing task demand problem of IoV users. Facing the situation that users choose local computing or offload the task to MEC servers, we use performance-price ratio strategy to reduce user cost, and the evaluation results show that the user cost of the algorithm is lower than the other three algorithms. The main contributions are as follows,

- We study the scenario in which the computing task is offloaded to RSU by the automatic driving vehicle, and consider the multi-MEC collaborative computing task to ensure that the task can be completed within the tolerable delay.
- We use network slicing to segment MEC server resources, so that one RSU can serve multiple users, and multiple RSUs can also cooperate with each other in the computing task.
- We propose a Performance-Price Ratio Task Scheduling (PPRTS) algorithm, which aims to complete the computing task within the maximum tolerable delay and reduces the cost of user. Compared with the other three algorithms, PPRTS always ensures the lowest cost of the user when the computing task is completed within the tolerable delay.

The rest of the paper is organized as follows. In Sect. 2, we introduce the system model and problem formulation. In Sect. 3 is our proposed solution strategy. In Sect. 4, we compare the performance of different algorithms and analyze the different parameters of PPRTS algorithm followed by the conclusion in Sect. 5.

2 System Model

In this section, we first introduce the application scenario and the slicing resources model. Next, we propose mobility and communication model to describe the condition of the vehicle and task. Then, under appropriate assumptions, we give the slicing resources pricing model and design the specific pricing function. Finally, we integrate the above three models to propose the optimization objectives and related constraints.

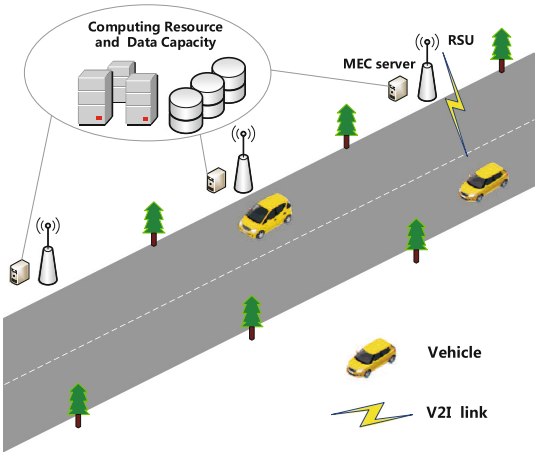

Fig. 1. Computational offloading of Multi-MEC collaboration.

2.1 Slicing Resources Model

As shown in Fig. 1, we consider that in the automatic driving scenario, some vehicles are driving on the straight road, and they can communicate with RSUs equipped with the MEC server through V2I link. We denote RSU set as $\mathcal{R} = \{1, 2, ..., N\}$. It is assumed that there is a certain distance between RSUs and the coverage is contiguous but not overlapping. Each RSU is equipped with the MEC server, which has limited data capacity and computing power. We mark the i-th RSU of vehicle connection as RSU_i, whose total data capacity is C_i and total computing power is A_i.

For vehicles, they have data capacity c_v, weak computing power a_v and low unit price p_v. Following [18], we use a triplet $\langle \delta, M, d \rangle$ to describe the task that the vehicle need to compute, where δ represents the complexity factor of the task, M represents the data size, and d represents maximum tolerable delay of the task. The computing tasks of vehicles are different in the form of text, voice or video, so the computing complexity factor of these tasks is different.

For RSUs, the computing resources and data capacity of multiple MEC servers are composed of network slices, which can be dynamically allocated to multiple vehicles. When the vehicle enters the coverage area of RSU, diversified logical private network services can be realized by accessing the network slicing. In this way, there is no need to build a physical private network and avoid the extra cost and waste of resources.

For the urgent computing tasks of vehicles, due to the weak local computing power, it may not be able to complete within the tolerable delay, which is an undesirable result for users. When a part of the tasks are offloaded to RSU and multiple MEC servers with strong computing power are used to process it simultaneously, the computing delay can be saved.

And the composition of the network slice of the RSU_i is

$$S_i = \{c_i, a_i, p_i\}, \tag{1}$$

where c_i and a_i represent data capacity and computing power of the slice respectively, and p_i represents the unit price of using this slice.

2.2 Mobility and Communication Model

There will be some complicated computing tasks in the process of vehicle driving to meet the safety, entertainment and other needs of passengers. According to the complexity factor δ, data size M and maximum tolerable delay d of the task, the computing task can be divided [19]. One part of the task m_v is computed locally, the other part of the task m_i is offloaded to RSU and computed by utilizing MEC servers slicing resources. The task local execution delay is as

$$t^v = \frac{\delta m_v}{a_v}, \tag{2}$$

where m_v is the data size computed locally by the vehicle, and a_v is the local computing power of the vehicle.

In the case of offloading the task to RSU and receiving the returned result, the vehicle location is different, and the vehicle mobility needs to be considered. Suppose there are T slots, set $\mathcal{T} = \{1, 2, ..., T\}$. At time t, we mark the coordinate of the vehicle as (x_t^v, y_t^v), and the RSU_i is (x_i, y_i), where $i \in \{1, 2, ..., N\}$. The distance between the vehicle and RSU_i can be calculated by Euclidean Distance as

$$d_i(t) = \sqrt{(x_t^v - x_i)^2 + (y_t^v - y_i)^2}, \forall i \in N. \tag{3}$$

And the channel gain between them is

$$g_i(t) = g_0 \rho^2 d_i(t)^{-\alpha_h}, \forall i \in N, \tag{4}$$

where g_0 is the channel gain at the reference distance, ρ is an exponential random variable with an mean value, d_i is the distance between the vehicle and the RSU_i, and α_h is the path loss index in the V2I link.

When K vehicles offload tasks to the same RSU through the shared channel, the signal-to-interference-plus-noise ratio (SINR) between the vehicle and RSU_i is

$$\gamma_i(t) = \frac{pg_i(t)}{\sigma^2 + \sum_{j=1}^{K-1} q_j h_j}, \forall i \in N, \tag{5}$$

where p is the transmission power of the current vehicle, g is the channel gain between the current vehicle and RSU; q is the transmission power of other vehicles, and h is the channel gain between other vehicles and RSU. And σ^2 is the power of additive white Gaussian noise.

Thus, the instantaneous data transmission rate between vehicle and RSU_i is

$$R_i(t) = Wlog_2\left(1 + \gamma_i(t)\right), \forall i \in N, \tag{6}$$

where W represents the channel bandwidth. The instantaneous data transmission rate of vehicle is measured once in each time slot, and the average transmission rate can be calculated as

$$\bar{R} = \frac{\sum_{t=1}^{T} R_i(t)}{T}, \forall i \in N. \tag{7}$$

We denote β^u and β^d as transmission overhead of uplink and downlink respectively. Then, the uplink t^u and downlink t^d transmission delay of vehicle remaining computing task offloaded to RSU are respectively represented as follows

$$t^u = \frac{\beta^u\left(M - m_v\right)}{\bar{R}}, \tag{8}$$

$$t^d = \frac{\beta^d\left(M - m_v\right)}{\bar{R}}. \tag{9}$$

Short distance optical fiber communication is used between RSUs, and the transmission delay between them can be ignored. After receiving the computing task of vehicle offloading, RSU_i will assign the task to the next RSUs, which will execute the computing task at the same time. So the task execution delay in MEC is

$$t^m = \max\left(\frac{\delta m_i}{a_i}\right), \forall i \in N, \tag{10}$$

where m_i is the data size computed by slice of the i-th MEC, and a_i is computing power of slice of the i-th MEC. After MEC servers cooperatively processes the data, the network slice transmits the task results from the nearest RSU to the vehicle. Thus, the total execution delay of the vehicle computing task is

$$t^{total} = \max\left(t^v, t^u + t^m + t^d\right). \tag{11}$$

We divide the computing task into two parts: one part is executed locally on the vehicle, the other part is offloaded to MEC servers on RSUs for execution. The two parts of the task are carried out at the same time, so the total delay depends on the part that takes more time.

2.3 Pricing Model

For vehicles, local processing of computing task requires energy consumption, and the unit price p_v is a low fixed value. For RSUs, MEC servers have different computing power. In industries with high fixed equipment cost and low marginal cost, dynamic pricing of resources can improve the enthusiasm of users to use resources and ensure the efficiency of resource utilization. Similar to [20], we consider the computing power and resource occupancy rate of MEC to ensure the user experience. Thus, we propose the unit price function of slicing resources as

$$P^{unit} = \lambda e^{\mu x} + \nu, \tag{12}$$

where x is related to computing resource occupancy rate of MEC. λ is the initial unit price of slicing resources. Its value is dynamic, and it is the proportion of the current MEC server and the largest server computing power. μ represents the degree of unit price change, which affects how fast the unit price changes with x. ν represents the minimum unit price provided by infrastructure provider. And λ and ν jointly determine the starting price of resources. All the above parameters are positive.

In order to complete the computing task, the cost of local processing on the vehicle is

$$P_v = p_v t^v. \tag{13}$$

Because multiple MEC slicing resources are used to cooperatively process the computing task, the processing cost of the task offloaded to RSUs is

$$P_m = \sum_{i=1}^{N} p_i^{unit} \frac{\delta m_i}{a_i}, \tag{14}$$

where p_i is the unit price of the i-th MEC slice according to (12), and $\frac{\delta m_i}{a_i}$ is the usage time of the i-th MEC slice. The total cost of computing task offloaded to RSUs is the sum of slicing resources costs.

By integrating the above three models, the user's computing task cost can be defined as a constrained optimization problem, as follows:

$$\begin{aligned} \textbf{P1:} \quad & \min \quad P_v + P_m \\ s.t. \quad & C1: m_v + \sum\nolimits_{i=1}^{N} m_i = M, \\ & C2: a_v + \sum\nolimits_{i=1}^{N} a_i \geqslant \frac{M}{d - t^u - t^d}, \\ & C3: t^{total} \leqslant d, \\ & C4: m_v \leqslant c_v, \\ & C5: m_i \leqslant c_i, \forall i \in N \end{aligned} \tag{15}$$

where $C1$ represents that the total the task assigned to vehicle and MEC servers is certain. $C2$ represents that the computing power of the vehicle and slices of

MEC servers can complete the computing task. $C3$ represents that the maximum delay for processing the computing task locally or MEC shall not exceed maximum tolerable delay of the task. $C4$ and $C5$ ensure that the data size of the task assigned to the vehicle or each slice does not exceed the data capacity of them, otherwise the task data will be lost.

3 Proposed Solution Strategy

In this section, we discuss the solution, and propose algorithm based on greedy algorithm to solve **P1** quickly.

Firstly, we analyze the trend of the pricing function in problem **P1**. The pricing function is a monotonically increasing function. Its characteristic is that the higher the occupancy rate of computing resource, the higher the unit price, and the larger the slope. Such a mechanism can reduce congestion caused by high resource occupancy rate, which is helpful to improve user experience.

Next, we consider a special case of problem **P1**: the vehicle and all MEC servers are isomorphic, so the computing power, data capacity and resources price of the vehicle and MEC servers are the same. For problem **P1**, the task can be divided into multiple items according to the data capacity of MEC servers, and the MEC server can be seen as a bin. Thus, the cost of user can be minimized by reasonably dividing the task, processing it locally or offloading it to the MEC servers. In this case, problem **P1** can be regarded as: minimizing the number of bins used by reasonably placing items.

In fact, each MEC server has different computing power, data capacity and resource price. In order to get more computing power at a cheaper price, we consider designing a greedy algorithm based on performance-price ratio strategy to reduce the cost of resource utilization.

Initialization: Recording the vehicle coordinates, judging which RSU the vehicle is in and select the RSU to offload. If the distance between the vehicle and RSU is less than the radius, it means that it is within its coverage, as follows

$$d_i(t) \leqslant R_r, \forall i \in N. \tag{16}$$

Stage 1: Selecting bins according to performance-price ratio and dividing the task by data capacity of each bin. We regard the vehicle and N MEC servers as a bins set $\mathcal{B} = \{b_1, ..., b_i, ..., b_{n+1}\}$, which have data capacity c_i, computing power a_i and unit price p_i. In order to minimize the cost of user, we try to use bin with high performance-price ratio under the premise of satisfying the constraints. The performance-price ratio is calculated as

$$r_i = \frac{a_i}{p_i}, \forall i \in N + 1. \tag{17}$$

Then direct performance-price strategy reorder the bin set $\mathcal{B}$ in descending order of the performance-price ratio.

Sometimes the amount of task data is too large, which will lead to large transmission delay. At this time, the offload data takes up a large part of the

Algorithm 1. Performance-Price Ratio Task Scheduling (PPRTS) Algorithm

Input:
Coordinates of RSUs and vehicle, RSU coverage radius Rr, vehicle speed Sv; complexity factor, data size and maximum tolerable delay of the task $\langle \delta, M, d \rangle$; data capacity, computing power and unit price of MEC servers slicing resources $\{c_i, a_i, p_i\}, \forall i \in N$ and the vehicle $\{c_v, a_v, p_v\}$.

Output:
Computation task scheduling scheme and minimum cost of the vehicle user $\mathcal{P}_v + \mathcal{P}_m$.

Determine which RSU the vehicle is located in according to (16), and the RSU to be offloaded is selected;
Create a bins set $\mathcal{B} = \{b_1, ..., b_i, ..., b_{n+1}\}$ of vehicle and MEC servers resources;
if direct performance-price strategy **then**
  Sort the set $\mathcal{B}$ in descending order according to (17);
else
  The local vehicle ranks first, and the rest sorts the set $\mathcal{B}$ in descending order according to (17);
end if
while the task is not fully divided **do**
  for $i \in N + 1$ **do**
    Check the data capacity c_i of b_i;
    $m_i \Leftarrow c_i$;
    Assign the task amount with data size m_i to b_i;
    $b_i \Leftarrow b_{i+1}$;
  end for
end while
The total delay t^{total} is calculated according to (11);
while $t^{total} > d$ **do**
  Assign the task amount $m_i \leqslant c_i$ that can be completed by b_{i+1} within d;
  $b_i \Leftarrow b_{i+1}$;
end while
According to the vehicle speed S_v and task uplink t^u and processing t^m delay, the new vehicle coordinates are calculated as $(x_t + S_v(t^u + t^m), y_t)$, and the RSU of the returned results is selected.
Generate task scheduling scheme and the user cost according to (13), (14).

time, and the final task is difficult to complete. In this case, the amount of data offloaded to MEC should be reduced, and the local computing resources should be utilized to reduce the transmission delay. Thus, for the task with large amount of data and high delay requirement, we adopt a local priority strategy, which makes maximum use of the local computing resources, and the rest part of the task is offloaded to the MEC servers according to the performance-price ratio.

The algorithm divides the task into many parts, checks the data capacity of $b1$, fills the data capacity of it, and loads the rest into $b2$ and $b3$ by analogy until the task assignment is completed. Next, the algorithm checks whether the delay of the scheme does not exceed maximum tolerable delay d. if not, the algorithm divides a part of the data m_{i+1} that can be completed in time d to b_{i+1}, and so

on; otherwise, the scheme is generated and the minimum user cost is calculated according to (13) and (14). Then, we compare the direct performance-price ratio strategy with the local priority strategy, and the task scheduling scheme adopts the one with lower user cost within the maximum tolerable delay.

Stage 2: Processing the computation task and returning the computation result. After formulating the task scheduling scheme, one part of the task is offloaded to MEC servers for cooperative processing, and the other part is processed locally. According to the vehicle speed and task uplink and processing delay, vehicle coordinates are calculated. Similar to (16), the RSU in coverage range is selected and the computation result is returned to the vehicle. And our proposed performance-price ratio task scheduling (PPRTS) algorithm is shown in Algorithm 1.

Finally, we analyze the time complexity of PPRTS algorithm. For the line 3 and 4 in Algorithm 1, specifically it will take $O(nlog_2(n))$ operations to sort all the bin in this line in descending order according to performance-price ratio. For lines 8–15 and 17–20 of the loop, each set will traverse once and then terminate within $O(n)$ operations in the worst case. In conclusion, the overall time complexity of PPRTS algorithm is $O(nlog_2(n))$.

4 Simulation Results

In this section, we present simulation results of the PPRTS algorithm and show its performance. In addition, compared with the other three schemes, we verify that the proposed scheme can complete the task within the maximum tolerable delay and the user cost is the lowest. Then, we analyze the influence of different parameters on the scheme through parameter experiment.

Table 1. Simulation parameters

Parameter	Value
Vehicle available data capacity c_v	100 GB
Vehicle computing capacity a_v	500 MHz
Local computing unit price p_v	100
Vehicle transmitting power p	37 dBm
Vehicle bandwidth W	100 MHz
Vehicle speed S_v	60 km/h
RSU coverage radius R_r	500 m
White Gaussian noise power σ^2	−118 dBm
Transmission overhead of uplink β^u and downlink β^d	1, 0.05
Number of vehicles K and MEC servers N	5, 3
Total data capacity of each MEC server	100 GB, 300 GB, 500 GB
Total computing power of each MEC server	1 GHz, 3 GHz, 5 GHz
Coefficient μ of pricing function	0.1
Minimum price ν of pricing function	10

4.1 Simulation Settings

We randomly simulate the usage of computing resource in MEC servers and conduct lots of experiments to verify the performance of PPRTS algorithm. Some parameters are listed in Table 1. It is worth noting that more MEC server cooperation is not always better, nor is shorter latency always better. Because our optimization goal is to complete the computing task within the maximum tolerable delay and minimize the cost of users. Too many MEC collaborative computing tasks can reduce the delay, but the cost of users will increase, which is not what users expect. According to the simulation experiment task data, the total computing resource of the three MEC servers are 1GHz, 3GHz and 5GHz respectively. In each experiment, the computing resource used and occupancy rate of three MEC servers are shown in Table 2. The tolerable delay of vehicle is strict, and low delay ensures normal driving. In the experiment, the total delay of computing task offloading to MEC, server processing task and result return is millisecond level. And the complexity factor, data size and maximum tolerable delay of vehicle computing task are shown in Table 3.

Table 2. Usage of MEC servers computing resource in each experiment

		MEC1	MEC2	MEC3
Set1	CR used (MHz)	772.10	1780.46	1646.73
	CR occupancy (%)	77.21	59.35	32.93
Set2	CR used (MHz)	615.18	771.56	4746.14
	CR occupancy(%)	61.52	25.72	94.92
Set3	CR used (MHz)	722.64	1332.38	1439.44
	CR occupancy (%)	72.26	44.41	28.79
Set4	CR used (MHz)	551.88	2535.63	3574.18
	CR occupancy (%)	55.19	84.52	71.48
Set5	CR used (MHz)	803.38	515.66	1932.10
	CR occupancy (%)	80.34	17.19	38.64

Table 3. Task status in each experiment

Task	Complexity factor δ	Data size $\mathcal{M}$	Tolerable delay d
Set1	1	12.5 MB	200 ms
Set2	1.5	25 MB	400 ms
Set3	1	50 MB	600 ms
Set4	2	37.5 MB	650 ms
Set5	2	50 MB	800 ms

4.2 Parametric Analysis

The unit price of resources is a noticeable factor, which determines the strategy of selecting MEC resources. Thus, we conduct simulation experiments to observe the impact of MEC server computing power and resource occupancy on unit price.

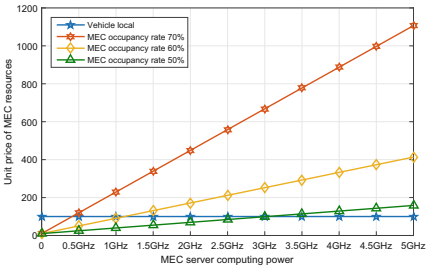

Fig. 2. Comparison of different MEC occupancy rates.

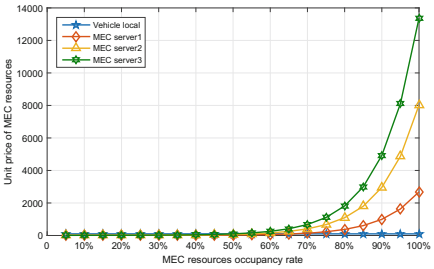

Fig. 3. Comparison of different MEC servers.

As shown in Fig. 2, the abscissa is the computing power of the MEC server, and the ordinate is the unit price of MEC resources. Taking the preset fixed vehicle local price as a reference, the influence of three MEC resource occupancy rates on resource unit price is compared. When the remaining computing power of MEC server is weak, the unit price is lower than vehicle local resources. With the enhancement of the remaining computing power of MEC server, the unit price also exceeds the vehicle. Under the same MEC computing capacity, the higher the occupancy rate, the higher the unit price, and the greater the growth rate of unit price. In this case, the MEC server with low resource occupancy and relatively strong computing power should be selected on the premise that the task can be completed within the tolerable delay.

Then, we compare the resource unit prices of heterogeneous MEC servers. Taking the preset fixed vehicle local price as a reference, three MEC servers use all the computing power, and the specific resources are shown in Table II. As shown in Fig. 3, the abscissa is the resource occupancy rate calculated by MEC, and the ordinate is the unit price of MEC resources. The unit price gap of each MEC server is small before 50%, and gradually widens after 60%, which is a protection mechanism. The high cost makes the user choose the server with relatively idle resources, ensuring the user experience and dispersing the server pressure of operator.

4.3 Scheme Comparison

We count the task delay, and use MEC resource pricing function under the same parameters to calculate the user cost of four task scheduling schemes in these

five Sets. The four task scheduling schemes are *PPRTS*, *PLRRO*, *All random offloading* and *Completely random*.

PPRTS uses our proposed Performance-Price Ratio Task Scheduling Algorithm. In other words, within the maximum tolerable delay, the scheme chooses the one with lower user cost between the direct performance-price ratio strategy and the local priority strategy. *PLRRO* means priority local residual random offloading. In this scheme, local computing task is considered first, and the rest of the task is randomly offloaded to multiple MEC servers. *All random offloading* represents that the task does not consider local computing at all, and all parts of the task are randomly offloaded to multiple MEC servers. *Completely random* represents a completely random proportion of the task being computed locally and offloaded to MEC servers.

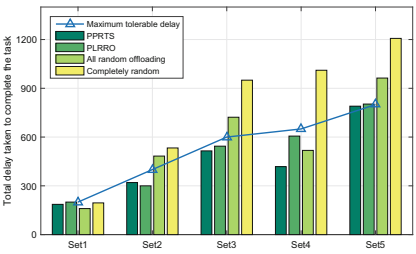

Fig. 4. Total delay of different schemes.

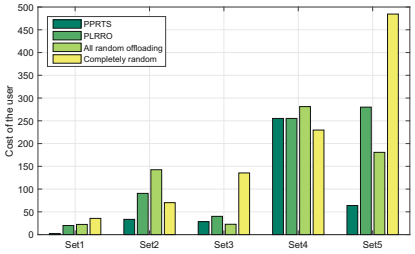

Fig. 5. Cost of different schemes.

The total delay of different schemes is shown in Fig. 4, and should not exceed the maximum tolerable delay. The abscissa is the MEC computing resource usage in five sets, and the ordinate is total delay of four schemes to complete the task. From the experimental results, not every scheme can complete the computing task within the maximum tolerable delay.

In these five Sets, *PLRRO* once, *All random offloading* three times, and *Completely random* four times failed to complete the computing task within the maximum tolerable delay. And Each experiment of *PPRTS* can complete the task within the tolerable delay. In Set 2, we find that the delay of *PLRRO* is lower than that of *PPRTS*, but it is not that the lower the delay is, the cheaper the price is. It may be that the *PLRRO* uses better computing resources to complete the computing task with lower delay, but the user cost is not necessarily the lowest. Our goal is to minimize the user cost within the tolerable delay of the task, so we only need to complete the task within the tolerable delay.

The user cost comparison of the four schemes is shown in Fig. 5. The abscissa is the MEC computing resource usage in these five sets, and the ordinate is user cost of four schemes to complete the task.

Due to the randomness, the user cost of *Completely random* is sometimes high and sometimes low, and the performance is unstable in these five Sets. If the task data is large and all parts of the task are offloaded to MEC, most of the delay is

spent on transmission, leaving less computing time, and the performance of *All random offloading* without considering local computing is not ideal. Although the cost of *All random offloading* is lowest in Set 3, it can not complete the task within the tolerance delay, so it is not desirable. Relatively speaking, *PLRRO* gives priority to the local cheap computing resources processing the task, so the user cost is lower than the two random schemes. However, MEC resources are relatively idle and cheap in Set 5, and the cost of preferentially computing task locally is high. *PPRTS* selects resources according to performance-price ratio, and the user cost is always the lowest among the four schemes.

5 Conclusion

In this paper, we study the cost of the user computing task for vehicle in autonomous driving scenario. In order to minimize the cost of user, we use network slicing to segment MEC server resources, and propose PPRTS algorithm, which offloads the computing task to multi-MEC servers according to performance-price ratio of slicing resources. Simulation results show that the algorithm can complete the task within the maximum tolerable delay, and effectively reduce the cost of user.

Acknowledgement. This work is partly supported by the National Natural Science Foundation of China (Nos. 61872044, 61902029), The Key Research and Cultivation Projects at Beijing Information Science and Technology University (No. 5211823411).

References

1. Dai, C., Liu, X., et al.: A low-latency object detection algorithm for the edge devices of IoV systems. IEEE Trans. Veh. Technol. **69**(10), 11169–11178 (2020)
2. Tang, X., Geng, Z., Chen, W.: Safety message propagation using vehicle-infrastructure cooperation in urban vehicular networks. In: Gao, H., Wang, X., Yin, Y., Iqbal, M. (eds.) CollaborateCom 2018. LNICST, vol. 268, pp. 235–251. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-12981-1_17
3. Lien, S., Hung, S., et al.: Low latency radio access in 3GPP local area data networks for V2X: stochastic optimization and learning. IEEE Internet Things J. **6**(3), 4867–4879 (2019)
4. Xiao, X., Li, Y., Xia, Y., Ma, Y., Jiang, C., Zhong, X.: Location-aware edge service migration for mobile user reallocation in crowded scenes. In: Gao, H., Wang, X., Iqbal, M., Yin, Y., Yin, J., Gu, N. (eds.) CollaborateCom 2020, Part I. LNICST, vol. 349, pp. 441–457. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-67537-0_27
5. Campolo C., Iera A., et al.: MEC support for 5G–V2X use cases through docker containers. In: IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–6 (2019)
6. Feng, J., Pei, Q., et al.: Dynamic network slicing and resource allocation in mobile edge computing systems. IEEE Trans. Veh. Technol. **69**(7), 7863–7878 (2020)
7. Liu, Y., Peng, M., et al.: Toward edge intelligence: multiaccess edge computing for 5G and internet of things. IEEE Internet Things J. **7**(8), 6722–6747 (2020)

8. Coronado E., Riggio R.: Flow-based network slicing: mapping the future mobile radio access networks. In: IEEE The 28th International Conference on Computer Communications and Networks (ICCCN), pp. 1–9 (2019)
9. Baek, B., Lee, J., et al.: Three dynamic pricing schemes for resource allocation of edge computing for iot environment. IEEE Internet Things J. **7**(5), 4292–4303 (2020)
10. Cardellini, V., Valerio, V.D., et al.: Game-theoretic resource pricing and provisioning strategies in cloud systems. IEEE Trans. Serv. Comput. **13**(1), 86–98 (2020)
11. Wang G., Feng G., et al.: Resource allocation for network slices in 5G with network resource pricing. In: IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (2017)
12. Wang, G., Feng, G., et al.: Reconfiguration in network slicing–optimizing the profit and performance. IEEE Trans. Netw. Serv. Manag. **16**(2), 591–605 (2019)
13. Liu, M., Liu, Y.: Price-based distributed offloading for mobile-edge computing with computation capacity constraints. IEEE Wirel. Commun. Lett. **7**(3), 420–423 (2018)
14. Liu, Y., Yu, F.R., et al.: Distributed resource allocation and computation offloading in fog and cloud networks with non-orthogonal multiple access. IEEE Trans. Veh. Technol. **67**(12), 12137–12151 (2018)
15. Wang, C., Liang, C., et al.: Computation offloading and resource allocation in wireless cellular networks with mobile edge computing. IEEE Trans. Wirel. Commun. **16**(8), 4924–4938 (2017)
16. Huang, C., Lin, T., et al.: Data dissemination of application service by using member-centric routing protocol in a platoon of internet of vehicle (IoV). IEEE Access **7**, 127713–127727 (2019)
17. Chen, C., Chen, L., et al.: Delay-optimized V2V-based computation offloading in urban vehicular edge computing and networks. IEEE Access **8**, 18863–18873 (2020)
18. Wang, Y., Lang, P., et al.: A game-based computation offloading method in vehicular multiaccess edge computing networks. IEEE Internet Things J. **7**(6), 4987–4996 (2020)
19. Lan Y., Wang X., et al.: Mobile-edge computation offloading and resource allocation in heterogeneous wireless networks. In: IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–6 (2019)
20. Liang Y., Chen X., et al.: Cooperative resource sharing strategy with eMBB cellular and C-V2X slices. In: IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS), pp. 716–721 (2020)