MSCET: A Multi-Scenario Offloading Schedule for Biomedical Data Processing and Analysis in Cloud-Edge-Terminal Collaborative Vehicular Networks

Zhichen Ni, Honglong Chen, *Senior Member, IEEE*, Zhe Li, Xiaomeng Wang,
Na Yan, Weifeng Liu, *Senior Member, IEEE*, and Feng Xia, *Senior Member, IEEE*

Abstract—With the rapid development of Artificial Intelligence (AI) and Internet of Things (IoTs), an increasing number of computation intensive or delay sensitive biomedical data processing and analysis tasks are produced in vehicles, bringing more and more challenges to the biometric monitoring of drivers. Edge computing is a new paradigm to solve these challenges by offloading tasks from the resource-limited vehicles to Edge Servers (ESs) in Road Side Units (RSUs). However, most of the traditional offloading schedules for vehicular networks concentrate on the edge, while some tasks may be too complex for ESs to process. To this end, we consider a collaborative vehicular network in which the cloud, edge and terminal can cooperate with each other to accomplish the tasks. The vehicles can offload the computation intensive tasks to the cloud to save the resource of edge. We further construct the virtual resource pool which can integrate the resource of multiple ESs since some regions may be covered by multiple RSUs. In this paper, we propose a Multi-Scenario offloading schedule for biomedical data processing and analysis in Cloud-Edge-Terminal collaborative vehicular networks called MSCET. The parameters of the proposed MSCET are optimized to maximize the system utility. We also conduct extensive simulations to evaluate the proposed MSCET and the results illustrate that MSCET outperforms other existing schedules.

Index Terms—Biomedical Data Processing and Analysis, Cloud-Edge-Terminal Collaborative Vehicular Networks, Optimization, Resource Allocation, Task Offloading.

✦

1 INTRODUCTION

THE rapid development of Artificial Intelligence (AI) [3], [14] and Internet of Things (IoTs) [1], [4], [15] has paved a way to realize modern applications in vehicles. For example, on-board AI plays an important role in the analysis of driver biomedical data [22], [29]. However, these applications are mostly computation intensive or delay sensitive while the computation resource of vehicles is relatively limited. This conflict may decline the Quality of Service (QoS) and eventually restrict the development of vehicular networks [19], [25]. Cloud computing can be used in such kind of scenario [10], [23], in which vehicles can offload tasks to resource-rich cloud servers via wireless network. However, the latency for the cloud computing may be too long, resulting in that some tasks cannot be accomplished in time.

Edge computing is a new paradigm and has attracted lots of attentions from both the research and industrial community. In contrast to cloud computing, edge computing brings computation services from the remote cloud to the network edge, enabling that some tasks can be quickly processed near the vehicles [12]. Recently, edge computing for vehicular networks has been studied, and some research works [18], [21] have been proposed.

Most of the traditional vehicular networks concentrate on Edge Servers (ESs) in Road Side Units (RSUs), while some tasks may be too complex for ESs to process. A potential solution is to combine cloud computing with edge computing, where the vehicles can offload the complex and delay-tolerant tasks to Cloud

Servers (CSs) [11]. Moreover, with the development of Network Function Virtualization (NFV) technology, the construction of virtual resource pool becomes feasible [13]. Some regions may be covered by multiple RSUs in the vehicular networks [27]. In such situation, the virtual resource pool can be utilized in the offloading schedule to integrate the resource of multiple ESs. And we further adopt Software Defined Network (SDN) to manage the virtual resource pool to improve its performance.

In this paper, we concentrate on investigating the problem of efficient and reliable multi-scenario offloading schedule for biomedical data processing and analysis in Cloud-Edge-Terminal collaborative vehicular networks. To this end, there are two challenges that need to be well addressed. The first one is the selection of vehicles' offloading targets, which can affect the optimization of the QoS utility. For instance, it will cause the overload of RSU if a numerous number of vehicles offload tasks to the same RSU, which may result in the timeout of tasks [28]. Thus, we adopt a matching method for the selection of each vehicle's offloading target and set a dynamic maximum connection limit for each RSU. The second challenge is how to obtain the reasonable offloading ratio and resource allocation to improve the system utility. To address these challenges, we consider a Cloud-Edge-Terminal collaborative vehicular network consisting of a CS, a set of RSUs, each of which is equipped with an ES, and a set of vehicles. Each vehicle has a computation task and can select servers to offload the task to. By jointly optimizing the offloading targets selection, offloading ratio and resource allocation, we obtain the near-optimal schedule to maximize the system utility.

The main contributions of this paper are the following:

- We model the offloading decision and resource allocation for Cloud-Edge-Terminal collaborative vehicular net-

- *Zhichen Ni, Honglong Chen, Zhe Li, Xiaomeng Wang, Na Yan and Weifeng Liu are with College of Control Science and Engineering, China University of Petroleum (East China), Qingdao 266580, P. R. China.*
- *Feng Xia is with the School of Science, Engineering and Information Technology, Federation University Australia, Ballarat, VIC 3353, Australia.*

 # Corresponding author: Honglong Chen. Email: chenhl@upc.edu.cn.

works, and consider both the profits and QoS in the system utility;

- To improve the processing efficiency of tasks, we construct the virtual resource pool to integrate the resource of multiple ESs and further introduce the SDN for the centralized management of the pool.
- We propose a Multi-Scenario offloading schedule for biomedical data processing and analysis tasks in Cloud-Edge-Terminal collaborative vehicular networks (MSCET), which optimizes the resource allocation, offloading targets and offloading ratio to maximize the system utility;
- We conduct extensive simulations to evaluate the performance of the proposed MSCET and the results illustrate that MSCET outperforms other existing schedules.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 presents the system model and problem formulation. We propose the MSCET schedule with theoretical analysis in Sections 4. In Section 5, we conduct the performance evaluation. Finally, this paper is concluded in Section 6.

2 RELATED WORK

Edge computing has been well studied in recent years and many research works have been proposed. Typically, some of them focused on how to make the reasonable offloading selections of servers [8], [35], while some others showed interest in optimizing the offloading ratio and resource allocation to maximize the system utility [30], [31]. Guo *et al.* [8] proposed a heuristic greedy offloading scheme for the mobile edge computation offloading problem in ultra-dense network and the schedule showed superior performance by conducting computation offloading over multiple mobile edge computing servers. For optimal service provisioning, Yu *et al.* [31] formulated an optimization problem to minimize the weighted sum of the service delay of all IoT devices and energy consumption by jointly optimizing the UAV position, communication and computing resource allocation, and task splitting decisions. Then they developed an efficient algorithm based on the successive convex approximation to obtain suboptimal solutions.

In order to address the conflict between the computation intensive or delay sensitive applications and resources-constrained vehicles, there are some research works on the combination of edge computing and vehicular networks in recent years [6], [17], [26]. Dai *et al.* [6] studied the resource allocation problem for a multi-user multi-server vehicular edge computing system and proposed a low-complexity algorithm to jointly optimize the server selection, offloading ratio and computation resource. The algorithm showed the superior performance in reducing the task processing delay while avoiding server overload. Liu *et al.* [17] considered the multiple vehicles' computation offloading problem in vehicular edge networks and formulated it as a multi-user computation offloading game problem. Then a distributed algorithm based on game theory was proposed to reduce the latency of the computation offloading of vehicles.

Some research works [16], [33] have studied on the integration of the SDN and vehicular edge computing networks. An SDN-enabled network architecture that integrated different types of access technologies was constructed in [16] to provide low-latency and high-reliability communication in mobile edge computing networks. Extensive numerical results showed that their proposed architecture has a significant improvement in the quality of user experience. Aiming to minimize the processing delay of the computation task of vehicles, Zhang *et al.* [33] proposed an SDN based load-balancing task offloading schedule in fiber-wireless (FiWi) enhanced vehicular edge computing networks, where SDN was introduced to provide supports for the centralized network and vehicle information management.

The delay of task processing can be decreased greatly with the assistance of edge computing, while some tasks may be too complex for edge servers to process. A potential solution is to combine the cloud computing with edge computing. Some works have focused on the cooperation between cloud and edge in recent years [24], [34]. Zhao *et al.* [34] presented a collaborative approach based on mobile edge computing and cloud computing in vehicular networks and further proposed a collaborative computation offloading and resource allocation optimization schedule to improve the system utility. To tackle the Problem of edge server overload, Shen *et al.* [24] proposed a dynamic task offloading method with minority game in cloud-edge computing. However, to simplify the model, they assumed that the tasks are offloaded to either the cloud server or the edge servers, which degraded the performance of cloud-edge cooperation networks.

The above works can improve the ability of the system for completing the complex tasks by utilizing the cooperation of cloud computing and edge computing. However, some of them only consider a relatively simple case such as binary decision. Moreover, they mainly focus on a single scenario and do not concentrate on the influence of the complex communication environment in vehicular networks. In this paper, we propose a multi-scenario offloading schedule for Cloud-Edge-Terminal cooperation vehicular networks such that each vehicle can adaptively select its schedule based on their environments.

3 SYSTEM MODEL AND PROBLEM FORMULATION

3.1 Network Framework

As shown in Fig. 1, we consider a Cloud-Edge-Terminal collaborative vehicular network consisting of a Cloud Server (CS), a set of Road Side Units (RSUs) $\mathcal{M} = \{E_1, E_2, \cdots, E_M\}$, and a set of vehicles $\mathcal{N} = \{V_1, V_2, \cdots, V_N\}$ that represent the terminal users. RSUs are randomly deployed in the area and each RSU is equipped with a resource limited Edge Server (ES). The coverage size of each RSU may be different due to the complex communication environment. Let $r = \{r_1, r_2, \cdots, r_M\}$ denote the coverage radius of each RSU. Some regions may be covered by multiple RSUs as the RSUs are randomly distributed, and we define them as the overlapping regions, such as region A shown in Fig. 1, and other regions as the general regions. We use $l_i = \{0, 1\}$ to represent the region that vehicle V_i is located in, where $l_i = 1$ represents that vehicle V_i is located in overlapping region, and $l_i = 0$ represents that vehicle V_i is located in general region.

In general region, we simplify the model to consider a uni-directional road, where M_G RSUs are deployed along the road, as shown in Fig. 2. While in overlapping region, we construct the virtual resource pool to integrate computation resource of the multiple distributed ESs using Network Function Virtualization (NFV) technology to improve the processing efficiency of tasks, and Software Defined Network (SDN) is adopted to provide centralized management for the resource pool. There are three parts in SDN architecture, i.e., control plane, data plane and application plane, the main components and responsibilities of which are as follows:
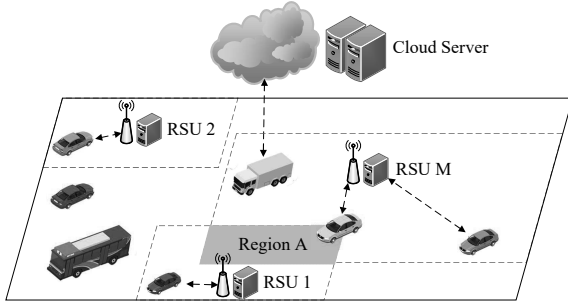
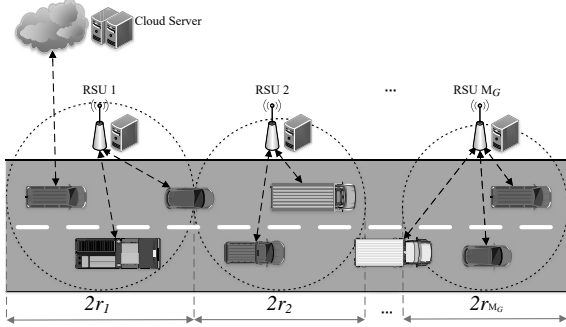Fig. 1. Task offloading in Cloud-Edge-Terminal collaborative vehicular networks.

Fig. 2. Task offloading in general region.

- Control Plane: It is made up of an SDN controller and RSU controllers, where SDN controller provides global management based on the information of RSU controllers, and RSU controllers are responsible for information collection on the resource of ESs and the tasks of vehicles. Moreover, they can also forward the offloading schedule of SDN controller.
- Data Plane: Data plane is a physical hardware layer that focuses on forwarding the data from RSUs and vehicles. It can be managed by either SDN controller or RSU controllers.
- Application Plane: It contains a series of hypervisors. In our constructed virtual resource pool, the main hypervisors consist of resource migration, OpenFlow management and offloading decision.

3.2 Task Model

Each vehicle V_i is running on the road at a speed of v_i and the safe distance between two vehicles is L. Vehicle V_i has a computation intensive or delay sensitive task that contains three characteristics $\{D_i, R_i, T_i^{max}\}$, where D_i (in bits) represents the size of task input data, R_i (in CPU cycles/bit) is the number of CPU cycles required to process 1-bit of the task, and T_i^{max} denotes the maximum tolerant delay to accomplish the task. Each task can be divided into three parts, which can be processed in the terminal, ES and CS in parallel. Let $\alpha_i = \{\alpha_{i,e}, \alpha_{i,c}\}$ $(0 \leq \alpha_{i,e} + \alpha_{i,c} \leq 1)$ denote the offloading ratio of vehicle V_i, then V_i offloads $\alpha_{i,c}D_i$ part of its task to the CS, offloads $\alpha_{i,e}D_i$ part of its task to the selected ES and processes the rest of its task locally. We denote $S_{i,j} = \{0, 1\}$ as the offloading decision of vehicle V_i on RSU, where $S_{i,j} = 1$ if vehicle V_i selects RSU E_j as its offloading target, and $S_{i,j} = 0$ otherwise.

3.3 Communication Model

There are two communication scenarios in our network, i.e., cloud communication and edge communication. For cloud communication, we assume that the data transmission rate is constant because of the long-distance between the vehicles and cloud [26]. Let $\rho_{i,CS}$ denote the data transmission rate between vehicle V_i and CS. The communication delay for cloud computing between vehicle V_i and CS, denoted as $T_{i,CS}^{comm}$, can be formulated as:

$$T_{i,CS}^{comm} = \frac{\alpha_{i,c}D_i}{\rho_{i,CS}} . \tag{1}$$

For edge computing in general region, each vehicle can establish a communication link with one of its nearby RSUs using vehicle to infrastructure (V2I) technology. Assume that the wireless channel is stable within a time slot (e.g., a few seconds), while it may change in different slots because of the mobility of vehicles [32]. According to Shannon Theory, the data transmission rate between vehicle V_i and RSU E_j, denoted as $\rho_{i,j}$, can be expressed as:

$$\rho_{i,j} = B \log \left(1 + \frac{P_i G_{i,j}}{\sigma + \sum_{i'=1, i' \neq i}^{N} P_{i'} G_{i',j}} \right), \tag{2}$$

where σ is white noise of the channel. B, P_i and $G_{i,j}$ are the bandwidth, the transmission power of vehicle V_i and the channel gain between vehicle V_i and RSU E_j respectively. Based on above analysis, the communication delay between vehicle V_i and RSU E_j for edge computing in general region, denoted as $T_{i,j}^{comm}$, is:

$$T_{i,j}^{comm} = \frac{\alpha_{i,e}D_i}{\rho_{i,j}}. \tag{3}$$

For edge computing in overlapping region, we denote $\rho_{i,P}$ as the data transmission rate between vehicle V_i and the virtual resource pool, and $T_{i,P}^{comm}$ as the communication delay between vehicle V_i and the virtual resource pool. The expressions of $\rho_{i,P}$ and $T_{i,P}^{comm}$ are similar to that of $\rho_{i,j}$ and $T_{i,j}^{comm}$, so we do not repeat them again.

By following the works in [5], [9], we ignore the download communication delay as the size of processed data is generally much smaller than that of input data when there is no server overload.

3.4 Computation Model

As vehicle V_i processes $(1 - \alpha_{i,e} - \alpha_{i,c})D_i$ part of its task locally and offloads the rest to the selected servers (i.e., ES and CS), we consider three different cases of the computation delay for ease of explanation.

3.4.1 Local Computing

Vehicle V_i processes $(1 - \alpha_{i,e} - \alpha_{i,c})D_i$ part of its task locally using its own resource. Different vehicles may have different resource, and the amount of resource can be obtained by offline measurement method [20]. Let $f_{i,V}$ (in CPU cycles/s) denote the computation resource of vehicle V_i, thus the local computation delay of V_i, denoted as T_i^{local}, can be expressed as:

$$T_i^{local} = \frac{(1 - \alpha_{i,e} - \alpha_{i,c})D_i R_i}{f_{i,V}} . \tag{4}$$

3.4.2 Edge Computing

For edge computing in general region, vehicle V_i offloads $\alpha_{i,e}D_i$ part of its task to RSU E_j. As each ES is resource limited, it is

necessary to allocate its computation resource. We denote $f_{i,j}$ (in CPU cycles/s) as the allocated resource of RSU E_j's ES to vehicle V_i. The computation delay for edge computing in general region when vehicle V_i offloads task to RSU E_j, denoted as $T_{i,j}^{comp}$, can be formulated as:

$$T_{i,j}^{comp} = \frac{\alpha_{i,e} D_i R_i}{f_{i,j}}, \tag{5}$$

$$s.t. \quad 0 \leq \sum_{i=1}^{N} S_{i,j} f_{i,j} \leq F_j \quad \forall j \in [1, \cdots, M_G], \tag{5a}$$

where F_j is the total resource of ES in RSU E_j.

For edge computing in overlapping region, vehicles do not need to select the offloading target as the ESs of all the RSUs are integrated as the virtual resource pool. We denote $f_{i,P}$ (in CPU cycles/s) as the allocated resource from resource pool to vehicle V_i. The computation delay for edge computing in overlapping region of vehicle V_i, denoted as $T_{i,P}^{comp}$, is:

$$T_{i,P}^{comp} = \frac{\alpha_{i,e} D_i R_i}{f_{i,P}}, \tag{6}$$

$$s.t. \quad 0 \leq \sum_{i=1}^{N} f_{i,P} \leq F_P, \tag{6a}$$

where F_P is the total resource of the virtual resource pool.

3.4.3 Cloud Computing

As the CS is resource-rich, it does not need to consider the resource allocation. We denote $f_{i,CS}$ (in CPU cycles/s) as the resource CS provides for vehicle V_i, then the cloud computation delay of vehicle V_i, denoted as $T_{i,CS}^{comp}$, is:

$$T_{i,CS}^{comp} = \frac{\alpha_{i,c} D_i R_i}{f_{i,CS}}. \tag{7}$$

3.5 Processing Delay

As mentioned above, each task is divided into three parts, which will be processed in the terminal, ES and CS in parallel. Thus, the task processing delay is dominated by the largest one of the local computing delay, ES processing delay and CS processing delay. Note that vehicle V_i will establish a communication link with CS only when it finishes the communication with RSU, which means that the processing delay of CS consists of three parts, i.e., the computation delay, the communication delay between vehicle and RSU and the communication delay between vehicle and CS. Based on above analysis, the processing delay of vehicle V_i, denoted as T_i, can be expressed as:

$$T_i = \begin{cases} \max\left\{T_i^{local}, T_{i,P}^{ES}, T_{i,P}^{CS}\right\}, & \text{if } l_i = 1; \\ \sum_{j=1}^{M} S_{i,j} \max\left\{T_i^{local}, T_{i,j}^{ES}, T_{i,j}^{CS}\right\}, & \text{otherwise}, \end{cases} \tag{8}$$

where $T_{i,P}^{ES} = T_{i,P}^{comm} + T_{i,P}^{comp}$, $T_{i,P}^{CS} = T_{i,P}^{comm} + T_{i,CS}^{comm} + T_{i,CS}^{comp}$, $T_{i,j}^{ES} = T_{i,j}^{move} + T_{i,j}^{comm} + T_{i,j}^{comp}$ and $T_{i,j}^{CS} = T_{i,j}^{move} + T_{i,j}^{comm} + T_{i,CS}^{comm} + T_{i,CS}^{comp}$. $T_{i,j}^{move}$ is the time vehicle V_i moves to the coverage of RSU E_j, which can be formulated as:

$$T_{i,j}^{move} = \left[\frac{x_j - x_i - r_j}{v_i}\right]^+, \tag{9}$$

where x_j, x_i represent the coordinates of RSU E_j and vehicle V_i respectively, and $[\cdot]^+$ denotes $\max\{0, \cdot\}$.

3.6 System Utility

The objective of this paper is to maximize the system utility of the platform. The platform rents servers to provide computation resource and utilizes the resource to process the tasks from vehicles. Apart from profits, QoS is also one of the important indicators for the system utility. Therefore, we design the system utility that considers both the profits and QoS.

For the Cloud-Edge-Terminal collaborative vehicular networks, the main factor that affects the QoS is the processing delay. Generally, QoS should monotonically decrease with the increase of the processing delay. According to [6], we choose the logarithmic function as the QoS model. Thus, the QoS utility for vehicle V_i, denoted as U_i^{QoS}, is:

$$U_i^{QoS} = \beta_Q \log\left(1 + \varepsilon - T_i\right), \tag{10}$$

where β_Q is the weight of QoS, and ε is a known value that guarantees $U_i^{QoS} > 0$. Thus, the system utility for vehicle V_i, denoted as U_i, is formulated as:

$$\begin{aligned} U_i &= \beta_P\left(c_v(\alpha_{i,e} + \alpha_{i,c})D_i R_i - c_s(f_i + f_{i,CS})\right) + U_i^{QoS} \\ &= \beta_P\left(c_v(\alpha_{i,e} + \alpha_{i,c})D_i R_i - c_s(f_i + f_{i,CS})\right) \\ &\quad + \beta_Q \log\left(1 + \varepsilon - T_i\right), \end{aligned} \tag{11}$$

where β_P is the weight of profits, c_v and c_s represent the unit price that system charges from vehicles for the task processing and the unit price that the system rents the servers respectively, f_i (in CPU cycles/s) denotes the allocated computation resource to vehicle V_i from ES or virtual resource pool, i.e.,

$$f_i = \begin{cases} f_{i,P}, & \text{if } l_i = 1; \\ f_{i,j}, & \text{otherwise}. \end{cases} \tag{12}$$

3.7 Problem Formulation

We formulate the offloading schedule as an optimization problem and aim to maximize the system utility. Let $S = \{S_{i,j}\}$ denote the selection decision on RSUs, $\alpha = \{\alpha_c, \alpha_e\}$ denote the offloading ratio decision of the vehicles, $f = \{f_i\}$ denote the resource allocation decision for the vehicles. Then, we formulate the offloading schedule problem as follows:

$$P1 : \max_{S,\alpha,f} \sum_{i=1}^{N} U_i, \tag{13}$$

$$s.t. \quad T_i \leq T_i^{max} \quad \forall i \in [1, \cdots, N]; \tag{13a}$$

$$0 \leq \alpha_{i,e} + \alpha_{i,c} \leq 1 \quad \forall i \in [1, \cdots, N]; \tag{13b}$$

$$0 \leq \alpha_{i,e} \leq 1 \quad \forall i \in [1, \cdots, N]; \tag{13c}$$

$$0 \leq \alpha_{i,c} \leq 1 \quad \forall i \in [1, \cdots, N]; \tag{13d}$$

$$S_{i,j} = \{0, 1\} \quad \forall i \in [1, \cdots, N], \ j \in [1, \cdots, M_G]; \tag{13e}$$

$$\sum_{j=1}^{M} S_{i,j} = 1 \quad \forall i \in [1, \cdots, N]; \tag{13f}$$

$$0 \leq \sum_{i=1}^{N} S_{i,j} f_{i,j} \leq F_j \quad \forall j \in [1, \cdots, M_G]; \tag{13g}$$

$$0 \leq \sum_{i=1}^{N} f_{i,P} \leq F_P, \tag{13h}$$

Constraint (13a) ensures that the task of vehicle V_i can be accomplished within the maximum tolerant delay T_i^{max}. Then Constraints

TABLE 1
List of notations

Symbols	Definitions
Sets:	
$\mathcal{M}$	Set of RSUs.
$\mathcal{N}$	Set of vehicles.
r	Set of RSUs' radius.
Decision Variables:	
α_i	Offloading ratio of vehicle V_i.
$S_{i,j}$	Offloading decision of vehicle V_i on RSU E_j.
f_i	Allocated computation resource to vehicle V_i from ES or virtual resource pool.
Functions:	
$T_{i,j}^{move}$	The time vehicle V_i moves to the coverage of RSU E_j.
T_i^{local}	Local computing delay of vehicle V_i.
$T_{i,CS}^{comm}$	Communication delay for cloud computing between vehicle V_i and CS.
$T_{i,j}^{comm}$	Communication delay for edge computing between vehicle V_i and RSU E_j.
$T_{i,P}^{comm}$	Communication delay for edge computing between vehicle V_i and the virtual resource pool.
$T_{i,CS}^{comp}$	Cloud computation delay of vehicle V_i.
$T_{i,j}^{comp}$	Computation delay for edge computing in general region when vehicle V_i offloads task to RSU E_j.
$T_{i,P}^{comp}$	Computation delay for edge computing in overlapping region of vehicle V_i.
T_i	Processing delay of vehicle V_i.
U_i^{QoS}	QoS utility for vehicle V_i.
U_i	System utility for vehicle V_i.
Parameters:	
l_i	The region that vehicle V_i is located in.
D_i	The size of task input data of vehicle V_i.
R_i	The number of CPU cycles required to process 1-bit of the task of vehicle V_i.
T_i^{max}	The maximum tolerant delay to accomplish the task of vehicle V_i.
$\rho_{i,CS}$	Data transmission rate between vehicle V_i and CS.
$\rho_{i,j}$	Data transmission rate between vehicle V_i and RSU E_j.
$\rho_{i,P}$	Data transmission rate between vehicle V_i and the virtual resource pool.
$f_{i,V}$	Local computation resource of vehicle V_i.
$f_{i,CS}$	Allocated computation resource to vehicle V_i from CS.
F_j	Total computation resource of ES in RSU E_j.
F_P	Total computation resource of resource pool.
β_P	Weight of profits.
β_Q	Weight of QoS.
c_v	Unit price that system charges from vehicles for the task processing.
c_s	Unit price that system rents the servers.
ε	A known value that guarantees $U_i^{QoS} > 0$.

(13b), (13c) and (13d) guarantee that vehicle V_i can only offload a ratio of its task. Constraints (13e) and (13f) show that vehicle V_i can select only one RSU as its offloading target. Constraints (13g) and (13h) ensure that the allocated computation resource to the vehicles is no more than the total resource. Moreover, Constraint (13e) show that the offloading decisions (i.e., S) are binary, while the objective function of Problem P1 is non-linear with respect to S. Thus, Problem P1 is a mixed integer non-linear programming problem, which is NP-hard [7].

For ease of reference, we list the frequently used notations in TABLE 1.

4 THE PROPOSED MSCET SCHEDULE

In this section, we introduce the proposed multi-scenario offloading schedule for biomedical data processing and analysis in the Cloud-Edge-Terminal collaborative vehicular networks (MSCET). As mentioned above, there are two different cases for the proposed MSCET schedule (i.e., overlapping region and general region), we will discuss each of them in details.

4.1 Decision in Overlapping Region

In overlapping region, the virtual resource pool is constructed to integrate computation resource, and all vehicles in the region offload tasks to the resource pool. Thus, S requires no consideration. However, the remaining decision vectors (i.e., α_c, α_e and f) of the optimization problem in Eq. (13) are highly coupled with each other, which makes it more difficult to solve the problem. Decoupling the vectors is one of the feasible solutions. Specially, we can fix two of them as constants and then calculate the third one. The two vectors we fixed can be updated based on the result of calculation. The process can be repeated until convergence.

4.1.1 Optimization of Offloading Ratio for CS

As α_c is relatively weakly correlated with other vectors, its value will be firstly determined. Due to the assumption that α_e and f are constants, the delay of edge computing is also a constant. Thus, T_i can be reformulated as:

$$T_i = \max \left\{ T_i^{local}, T_{i,P}^{comm} + T_{i,CS}^{comm} + T_{i,CS}^{comp} \right\}. \qquad (14)$$

Since T_i is affected by but non-differentiable with respect to α_c, we first approximate T_i as:

$$\begin{aligned} T_i &\leq T_i^{local} + T_{i,P}^{comm} + T_{i,CS}^{comm} + T_{i,CS}^{comp} \\ &= \frac{\left(1 - \alpha_{i,e} - \alpha_{i,c}\right)D_i R_i}{f_{i,V}} + \frac{\alpha_{i,e} D_i}{\rho_{i,P}} + \frac{\alpha_{i,c} D_i}{\rho_{i,CS}} + \frac{\alpha_{i,c} D_i R_i}{f_{i,CS}} \\ &= \alpha_{i,c} \omega_{i,c} + \delta_{i,c}, \end{aligned} \qquad (15)$$

where $\omega_{i,c} = \frac{D_i}{\rho_{i,CS}} + \frac{D_i R_i}{f_{i,CS}} - \frac{D_i R_i}{f_{i,V}}$, $\delta_{i,c} = \frac{\alpha_{i,e} D_i}{\rho_{i,P}} + \frac{(1 - \alpha_{i,e})D_i R_i}{f_{i,V}}$.

Let T_i be the value of its upper bound, which represents the worst case. By substituting Eq. (15) into Problem P1, it can be reformulated as:

$$P2.1 : \max_{\alpha_c} \sum_{i=1}^{N} \beta_P \big(c_v(\alpha_{i,c} + \alpha_{i,e})D_i R_i - c_s(f_{i,P} + f_{i,CS}) \big)$$

$$+ \beta_Q \log \big(1 + \varepsilon - \alpha_{i,c} \omega_{i,c} - \delta_{i,c} \big), \qquad (16)$$

$$s.t. \quad \alpha_{i,c} \omega_{i,c} + \delta_{i,c} \leq T_i^{max} \quad \forall i \in [1, \cdots, N]; \qquad (16a)$$

$$0 \leq \alpha_{i,c} \leq 1 - \alpha_{i,e} \quad \forall i \in [1, \cdots, N]. \qquad (16b)$$

Lemma 1. *Problem P2.1 is a convex optimization problem.*

Proof. The second-order derivative of $U(\alpha_c)$ with respect to α_c is:

$$\frac{\partial^2 U(\alpha_c)}{\partial \alpha_c^2} = -\frac{\beta_Q \omega_{i,c}^2}{\left(1 + \varepsilon - \alpha_{i,c}\omega_{i,c} - \delta_{i,c}\right)^2 \ln 2} \leq 0, \qquad (17)$$

Thus the objective function $U(\alpha_c)$ is convex. Combining with the linear convex Constraints (16a) and (16b), Problem P2.1 is a convex optimization problem according to the definition of convex optimization problem [2]. □

According to Lemma 1, Problem P2.1 is a convex optimization problem. Here we use genetic algorithm to solve it.

4.1.2 Optimization of Computation Resource

After determination of α_c, we need to update the values of α_e and f. Based on the above decoupling method, the formulation of computation resource optimization under given α_c and α_e is:

$$P2.2 : \max_{f} \sum_{i=1}^{N} \beta_P \big(c_v(\alpha_{i,c} + \alpha_{i,e})D_i R_i - c_s(f_{i,P} + f_{i,CS})\big)$$

$$+ \beta_Q \log\left(1 + \varepsilon - T_i\right), \qquad (18)$$

$$s.t. \quad \sum_{i=1}^{N} f_{i,P} \leq F_P; \qquad (18a)$$

$$\frac{\alpha_{i,e}D_i}{\rho_{i,P}} + \frac{\alpha_{i,e}D_i R_i}{f_{i,P}} \leq T_i^{max} \quad \forall i \in [1, \cdots, N]. \quad (18b)$$

Note that the local computing delay and cloud computing delay are all constants under given α_c and α_e, we update the Constraint (13a) as Constraint (18b).

The objective function $U(f)$ in Problem P2.2 is convex since $\frac{\partial^2 U(f)}{\partial f^2} = -\frac{2\beta_Q \alpha_{i,e}D_i R_i}{f_{i,P}^3\left(1 + \varepsilon - \frac{\alpha_{i,e}D_i}{\rho_{i,P}} - \frac{\alpha_{i,e}D_i R_i}{f_{i,P}}\right)\ln 2}$

$-\frac{\beta_Q(\alpha_{i,e}D_i R_i)^2}{f_{i,P}^4\left(1 + \varepsilon - \frac{\alpha_{i,e}D_i}{\rho_{i,P}} - \frac{\alpha_{i,e}D_i R_i}{f_{i,P}}\right)^2 \ln 2} \leq 0,$ and Constraints (18a) and (18b) are linear convex. According to Lemma 1, Problem P2.2 is a convex optimization problem and can be solved by interior point method. The first step of interior point method is to construct a penalty function, which can be expressed as:

$$\varphi\left(f, r^{(k)}\right) = U\left(f\right)$$

$$- r_1^{(k)} \sum_{i=1}^{N} \frac{1}{\frac{\alpha_{i,e}D_i}{\rho_{i,P}} + \frac{\alpha_{i,e}D_i R_i}{f_{i,P}} - T_i^{max}} \qquad (19)$$

$$- r_2^{(k)} \frac{1}{\sum_{i=1}^{N} f_{i,P} - F_P},$$

where $r^{(k)}$ are penalty factors and obey the law of decline, i.e., $r^{(0)} > r^{(1)} > r^{(2)} > \cdots > r^{(k)} > r^{(k+1)} > \cdots > 0$ and $\lim_{k \to \infty} r^{(k)} = 0$.

The second step is setting $\frac{\partial \varphi}{\partial f} = 0$ to find the extreme point f^* of the penalty function. Then substitute f^* into the condition of convergence $\| f^*(r^k) - f^*(r^{k-1}) \| \leq \varepsilon_1$, where ε_1 is the threshold that we set in advance. If f^* satisfies the above condition, the process will be finished, and we can obtain $f = f^*$. Otherwise, let $k = k + 1$ and update $r^{(k)}$ continuously until satisfying the above condition.

4.1.3 Optimization of Offloading Ratio for ES

Similar to the optimization of offloading ratio for CS, T_i can be approximated as:

$$T_i \leq \frac{T_i^{local} + T_{i,P}^{comm} + T_{i,P}^{comp} + T_{i,P}^{comm} + T_{i,CS}^{comm} + T_{i,CS}^{comp}}{\lambda}$$

$$= \frac{\left(1 - \alpha_{i,e} - \alpha_{i,c}\right)D_i R_i}{\lambda f_{i,V}} + \frac{\alpha_{i,e}D_i}{\lambda \rho_{i,P}} + \frac{\alpha_{i,e}D_i R_i}{\lambda f_{i,P}}$$

$$+ \frac{\alpha_{i,e}D_i}{\lambda \rho_{i,P}} + \frac{\alpha_{i,c}D_i}{\lambda \rho_{i,CS}} + \frac{\alpha_{i,c}D_i R_i}{\lambda f_{i,CS}} \qquad (20)$$

$$= \alpha_{i,e}\omega_{i,e} + \delta_{i,e},$$

where $\omega_{i,e} = \frac{2D_i}{\lambda \rho_{i,P}} + \frac{D_i R_i}{\lambda f_{i,P}} - \frac{D_i R_i}{\lambda f_{i,V}}$, $\delta_{i,e} = \frac{\alpha_{i,c}D_i}{\lambda \rho_{i,CS}} + \frac{(1 - \alpha_{i,c})D_i R_i}{\lambda f_{i,V}} + \frac{\alpha_{i,c}D_i R_i}{\lambda f_{i,CS}}$, and λ is a known value to adjust the approximation degree. Then we can reformulate Problem P1 as:

$$P2.3 : \max_{\alpha_e} \sum_{i=1}^{N} \beta_P \big(c_v(\alpha_{i,c} + \alpha_{i,e})D_i R_i - c_s(f_{i,P} + f_{i,CS})\big)$$

$$+ \beta_Q \log\left(1 + \varepsilon - \alpha_{i,e}\omega_{i,e} - \delta_{i,e}\right), \qquad (21)$$

$$s.t. \quad 0 \leq \alpha_e \leq 1 - \alpha_c \quad \forall i \in [1, \cdots, N]; \qquad (21a)$$

$$\alpha_{i,e}\omega_{i,e} + \delta_{i,e} \leq T_i^{max} \quad \forall i \in [1, \cdots, N]. \qquad (21b)$$

The objective function $U(\alpha_e)$ of Problem P2.3 is convex since $\frac{\partial^2 U(\alpha_e)}{\partial \alpha_e^2} = -\frac{\beta_Q \omega_{i,e}^2}{\left(1 + \varepsilon - \alpha_{i,e}\omega_{i,e} - \delta_{i,e}\right)^2 \ln 2} \leq 0$, and Constraints (21a), (21b) are linear convex, thus the Problem P2.3 is a convex optimization problem according to Lemma 1. The Lagrangian method [26] is adopted to solve this problem. The Lagrangian function can be expressed as:

$$\mathcal{L}_i(\alpha_e, \zeta, \eta, \theta) = \beta_P \big(c_v(\alpha_{i,c} + \alpha_{i,e})D_i R_i - c_s(f_{i,P} + f_{i,CS})\big)$$

$$+ \beta_Q \log\left(1 + \varepsilon - \alpha_{i,e}\omega_{i,e} - \delta_{i,e}\right)$$

$$+ \zeta_i\big(\alpha_{i,e}\omega_{i,e} + \delta_{i,e} - T_i^{max}\big) \qquad (22)$$

$$+ \eta_i\big(\alpha_{i,e} + \alpha_{i,c} - 1\big) + \theta_i\big(0 - \alpha_{i,e}\big),$$

where ζ, η and θ are Lagrangian multipliers. According to KKT condition [2], the following conditions must be satisfied:

$$\begin{cases} \zeta_i\big(\alpha_{i,e}\omega_{i,e} + \delta_{i,e} - T_i^{max}\big) = 0; \\ \eta_i\big(\alpha_{i,e} + \alpha_{i,c} - 1\big) = 0; \\ \theta_i\big(0 - \alpha_{i,e}\big) = 0; \\ \zeta_i > 0, \ \eta_i \geq 0, \ \theta_i \geq 0; \\ \frac{\partial \mathcal{L}_i}{\partial \alpha_{i,e}} = 0. \end{cases} \qquad (23)$$

As mentioned in Constraint (21a), we know that $0 \leq \alpha_{i,e} \leq 1 - \alpha_{i,c}$. Thus, it is simple to solve $\eta_i(\alpha_{i,e} + \alpha_{i,c} - 1) = 0$ and $\theta_i(0 - \alpha_{i,e}) = 0$. The related discussion is as follows:

Case 1: $\eta_i > 0$ and $\theta_i > 0$. We get $\alpha_{i,e} = 0$ and $\alpha_{i,c} = 1$, i.e., vehicle V_i offloads the whole task to the CS.

Case 2: $\eta_i > 0$ and $\theta_i = 0$. We get $\alpha_{i,e} = 1 - \alpha_{i,c}$, i.e., vehicle V_i offloads the whole task to its selected ES and CS.

Case 3: $\eta_i = 0$ and $\theta_i > 0$. We get $\alpha_{i,e} = 0$, i.e., vehicle V_i

offloads $\alpha_{i,c}D_i$ part of its task to the CS and processes the rest locally.

Case 4: $\eta_i = 0$ and $\theta_i = 0$. We cannot get the unique solution of $\alpha_{i,e}$. Then it is necessary to utilize other conditions, since $\zeta_i > 0$, we can get:

$$\alpha_{i,e}\omega_{i,e} + \delta_{i,e} - T_i^{max} = 0$$
$$\Rightarrow \quad \alpha_{i,e} = \frac{T_i^{max} - \delta_{i,e}}{\omega_{i,e}} . \tag{24}$$

Then ζ_i can be calculated as:

$$\frac{\partial \mathcal{L}_i}{\partial \alpha_{i,e}} = \beta_P c_v D_i R_i - \frac{\beta_Q \omega_{i,e}}{\left(1 + \varepsilon - \alpha_{i,e}\omega_{i,e} - \delta_{i,e}\right)\ln 2} + \zeta_i \omega_{i,e} = 0$$
$$\Rightarrow \quad \zeta_i = \frac{\beta_Q}{\left(1 + \varepsilon - T_i^{max}\right)\ln 2} - \frac{\beta_P c_v D_i R_i}{\omega_{i,e}} \tag{25}$$

And we can get α_i as follows:

$$\alpha_{i,e} = \begin{cases} \dfrac{T_i^{max} - \delta_{i,e}}{\omega_{i,e}}, & \text{if } \eta_i = 0 \text{ and } \theta_i = 0; \\ 1 - \alpha_{i,c}, & \text{if } \eta_i > 0 \text{ and } \theta_i = 0; \\ 0, & \text{otherwise.} \end{cases} \tag{26}$$

4.2 Decision in general region

Different from overlapping region, each vehicle needs to select one RSU as its offloading target in general region. Therefore, it is necessary to calculate S. The matching method is adopted to get the optimal selection. Determining the appropriate matching weight between vehicles and RSUs is one of the challenges. The processing delay for edge computing in general region includes the moving time, which is affected by the distance between vehicles and RSUs. Thus, we use the distance as the weight of matching method.

The first step is to construct a weighted bipartite graph $\mathcal{G}(\mathcal{U}, \mathcal{V})$ to establish the relation between vehicles and RSUs. The set $\mathcal{U}$ represents the vehicles in general region, and the set $\mathcal{V}$ represents the RSUs along the unidirectional road.

The second step is to utilize Kuhn-Munkres algorithm to get the minimum weight matching after which S can be obtained. Specifically, $S_{i,j} = 1$ if vehicle V_i selects RSU E_j as its offloading target, and $S_{i,j} = 0$ otherwise.

According to the selection decision S, there may be some vehicles that need to move to the communication coverage of their selected RSUs. Cloud-Terminal cooperation is utilized for the movement to reduce the total task processing delay. Due to the fact that ESs are resource-limited, vehicle V_i will offload as high ratio of its task as possible to CS to save the resource of ESs. Thus, we set $T_{i,CS}^{comm^*} = T_{i,j}^{move}$, which represents the maximum offloading ratio for CS. Then the task characteristics of vehicle V_i can be updated as:

$$D_i^* = D_i - \frac{T_{i,j}^{move} f_{i,V}}{R_i} - T_{i,j}^{move} \rho_{i,CS}; \tag{27}$$
$$T_i^{max^*} = T_i^{max} - T_{i,j}^{move}. \tag{28}$$

The decisions of α_e, α_c and f of the new optimization problem are similar to the decisions in overlapping region, so we do not repeat them again.

4.3 The MSCET Schedule Algorithm

The pseudo of MSCET schedule algorithm is shown in Algorithm 1. Vehicle V_i will send the message about the characteristics $\{D_i, R_i, T_i^{max}\}$ of its task to the system when it cannot process the task in time. And then the system calculates the offloading schedule based on the messages from all the vehicles in the area. Specifically, there are two cases according to the location of each vehicle. In general region, each vehicle needs to select one RSU as its offloading target by Kuhn-Munkres matching method while in overlapping region the RSU selection is unnecessary. Then, Cloud-Terminal cooperation is utilized to improve the system utility during the moving time in general region. After that, Cloud-Edge-Terminal cooperation is introduced to further improve the utility. Last but not the least, the genetic algorithm, interior point method and KKT condition are adopted to optimize α_c, f and α_e until getting the final result.

The complexities of initialization, Phase 1 and Phase 2 are $O(3N + 3)$, $O(N^4)$ and $O(3N)$ respectively. While for k iterations, the complexity of the inner loop is $O(k(2N + 1))$. Then the total complexity of middle loop, for z iterations, is $O(z(k(2N + 1) + N))$. After that, for s iterations, the complexity of Phase 3 is $O(sN^2(z(k(2N+1) + N)))$. Considering the dominant term, the time complexity of Algorithm 1 is $O(N^4 + 2skzN^3)$.

Algorithm 1 MSCET Schedule Algorithm

Input: D_i: the size of task input data of vehicle V_i;
        R_i: the number of CPU cycles required to process
           1-bit of the task of vehicle V_i;
        T_i^{max}: the maximum tolerant delay of vehicle V_i's task.
Output: Offloading schedule consisting of S, α_c, α_e and f.
1: Initialization:
    Set initial feasible solution $\alpha_e^{(0)}$, $\alpha_c^{(0)}$, $f^{(0)}$;
    Set iteration number $s = 1$, $z = 1$, $k = 1$;
2: **if** $l_i = 0$ **then**
3:    /*Phase 1: Selection of RSU*/
4:    Get the minimum weight matching and $S_{i,j}$;
5:    /*Phase 2: Cloud-Terminal Cooperation*/
6:    Let $T_{i,CS}^{comm^*} = T_{i,j}^{move}$ and update D_i and T_i^{max} based on
     Eq. (27) and Eq. (28);
7: **end if**
8: /*Phase 3: Cloud-Edge-Terminal Cooperation*/
9: /*Phase 3.1: Optimization of offloading ratio for CS*/
10: **repeat**
11:    Utilize genetic algorithm to optimize $\alpha_{i,c}$;
12:    $s = s + 1$;
13:    /*Phase 3.2: Optimization of offloading ratio for ES*/
14:    **repeat**
15:      /*Phase 3.3: Optimization of computation resource*/
16:      **repeat**
17:        Construct the penalty function $\varphi\left(f, r^{(k)}\right)$ based on
         Eq. (19) and set $\dfrac{\partial \varphi}{\partial f} = 0$ to find the extreme point f^*
         of the penalty function;
18:        Substitute f^* into the convergence condition
         $\| f^*(r^k) - f^*(r^{k-1}) \| \le \varepsilon_1$ to verify its feasibility;
19:        $k = k + 1$ and update $r^{(k)}$;
20:      **until** Satisfaction
21:      Calculate $\alpha_{i,e}^{(z)}$ based on Eq. (26);
22:      $z = z + 1$;
23:    **until** Convergence
24: **until** Convergence

---

5 PERFORMANCE EVALUATION

In this section, we evaluate the proposed multi-scenario offloading schedule (MSCET) and compare it with two benchmark schedules and two different modes. We first describe the simulation settings, and then show the schedules' reliability. Afterwards we conduct the performance comparisons to illustrate the effectiveness of the proposed MSCET.

5.1 Simulation Settings

As mentioned, there are two scenarios in the vehicular network (i.e., overlapping region and general region). In general region, we consider a 250-meter unidirectional road, along which 5 RSUs are deployed. Each RSU is equipped with an ES, which has the computation resource of 0.5 GHz. While in overlapping region, the virtual resource pool is constructed to integrate resource of ESs, and the total resource of each virtual resource pool is 2.5 GHz. There are some vehicles running at a speed of 40 km/h. Each vehicle V_i has a computation-intensive or delay-sensitive task with three characteristics $\{D_i, R_i, T_i^{max}\}$. We set D_i and T_i^{max} to be randomly distributed in the range of $[10, 15]$MB and $[4, 6]$s respectively. The local computation resource of each vehicle is 12.5 MHz.

For performance comparison, we consider the following two benchmarks schedules for Cloud-Edge-Terminal cooperation and two cooperation modes using our proposed MSCET schedule.

1) Given Ratio and Resource based Selection optimization (SGRR): The schedule selects the optimal offloading target for each vehicle, under a given offloading ratio and computation resource.

2) Nearby: The schedule optimizes computation resource and offloading ratio, after which the vehicles offload their tasks to nearby RSUs.

3) Cloud-Terminal: Each task is jointly processed in the terminal and cloud server.

4) Edge-Terminal: Each task is jointly processed in the terminal and edge server.

5.2 Convergence

To verify the reliability of the proposed MSCET schedule, we evaluate the convergence of algorithm under different initial points. The algorithm can converge only if the inner loop can converge, thus the convergence of inner loop is omitted here and only the convergence of algorithm is plotted, i.e. the convergence of system utility. Fig. 3 shows that the system utility can converge within 6 iterations and almost to the same value under different initial points, indicating the reliability of the proposed MSCET schedule.

5.3 Impact of RSUs' Coverage Size

To compare the scalability of different schedules, we study the impact of the radius of RSUs' coverage on system utility in Fig. 4. The results illustrate that the system utility of the Nearby will be decreased with the increasing radius of RSUs' coverage, while the coverage size has no impact on the system utility of the other two schedules, which means that the two schedules have a better scalability. The reason is that both the MSCET and SGRR consider the offloading selection optimization, making them not sensitive to the increase of coverage size. However, for the Nearby, with the expansion of RSUs' coverage, the increasing number of vehicles
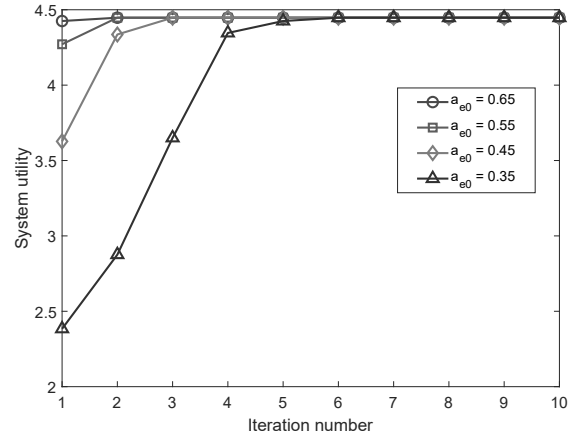

Fig. 3. Convergence of system utility under different initial points.
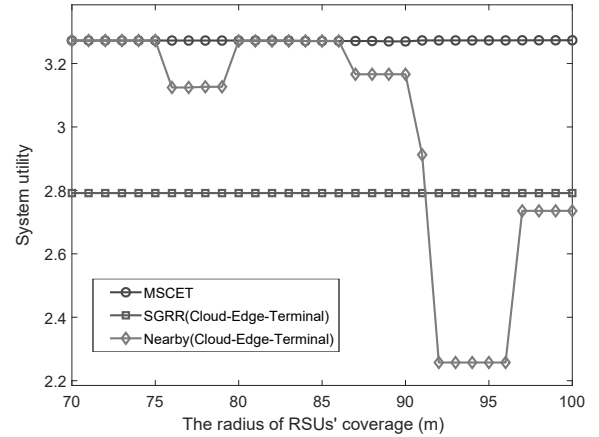

Fig. 4. Performance comparison of system utility under different sizes of RSUs' coverage.

in RSUs' coverage may cause that more vehicles connect to the same RSU, which may lead to the overload of RSU. In this case some tasks cannot be processed in the current time slot, and the system utility may become lower. Besides, the performance of the Nearby may fluctuate when some vehicles are located in the boundary of RSUs, shown as the fluctuation in Fig. 4. Moreover, the MSCET outperforms the SGRR due to its joint optimization of resource allocation and offloading selection.

5.4 Advantages of Cloud-Edge-Terminal Cooperation

In this subsection, we compare the performance of the proposed MSCET with different cooperation modes in general region. Fig. 5 shows the impact of the increasing number of vehicles on the system utility between different cooperation modes. The results illustrate that with the increasing number of vehicles, the utility under all the three modes will increase. This is because that the increasing number of vehicles will produce more tasks that need to be processed. And Cloud-Terminal cooperation performs a bit worse than other modes due to the long communication delay between CS and vehicles. Furthermore, the performance of MSCET is similar to that of Edge-Terminal when there are fewer vehicles. While MSCET outperforms Edge-Terminal when the number of vehicles exceeds a certain threshold. The reason behind is that the proposed MSCET is a collaborative offloading
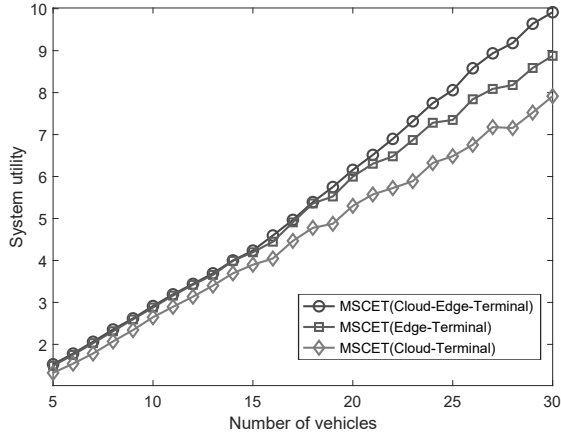
Fig. 5. Performance comparison of system utility under different cooperation modes with the increase of vehicles in general region.

illustrate that the MSCET outperforms existing schedules. However, some assumptions in our proposed MSCET are based on ideal condition and there may be some differences with the real scenario. In the future, we intend to extend the MSCET schedule to be applicable for the real-world scenario with implementation.

schedule combining ESs and CS and the tasks can be processed locally and in servers (i.e., ESs and CS) simultaneously. In other words, the tasks can be reasonably offloaded to reduce the impact of the resource-limited ESs. Moreover, the proposed MSCET can make full use of the moving time to process the tasks by utilizing Cloud-Terminal cooperation while Edge-Terminal can only utilize local resource to process the tasks until reaching the coverage of the selected RSUs.

5.5 Advantages of virtual resource pool

To illustrate the benefits of virtual resource pool, we compare the performance of MSCET with and without the virtual resource pool under different amount of resource. Fig. 6 illustrates that the performance of MSCET with or without the virtual resource pool is similar when the resource in overlapping region is abundant. However, the performance of MSCET without the pool will be worse when the resource is limited and the performance difference between MSCET with and without the pool increases with the decrease of resource. As the virtual resource pool integrates the resource from multiple distributed ESs. When some ESs are resource-limited, the pool can invoke the resource from other ESs to cooperate on the tasks. In contrast, ESs that without the assistance of the pool can only utilize their own resource to process the tasks and can be easily affected by the amount of the resource.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we study the task offloading problem for vehicular networks and propose MSCET, a multi-scenario offloading schedule for biomedical data processing and analysis tasks in Cloud-Edge-Terminal collaborative vehicular networks, which integrates the Cloud-Edge-Terminal cooperation and virtual resource pool to improve the efficiency and reliability. We consider both the profits and QoS as the system utility, and formulate the system utility maximization problem. We then show that the problem is NP-hard and present an algorithm consisting of the Kuhn-Munkres matching method, genetic algorithm, interior point method and KKT condition to obtain efficient solutions. The parameters of the proposed MSCET are optimized to maximize the system utility. Finally, we conduct extensive simulations to evaluate the performance of the proposed MSCET and the simulation results

REFERENCES

[1] X. Ai, H. Chen, K. Lin, Z. Wang, and J. Yu, "Nowhere to Hide: Efficiently Identifying Probabilistic Cloning Attacks in Large-Scale RFID Systems," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 714–727, 2021.

[2] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge university press, 2004.

[3] H. Chen, S. Wang, N. Jiang, Z. Li, N. Yan, and L. Shi, "Trust-aware Generative Adversarial Network with Recurrent Neural Network for Recommender Systems," *International Journal of Intelligent Systems*, vol. 36, pp. 778–795, 2021.

[4] H. Chen, Z. Wang, F. Xia, Y. Li, and L. Shi, "Efficiently and Completely Identifying Missing Key Tags for Anonymous RFID Systems," *IEEE Internet of Things Journal*, vol. 5, pp. 2915–2926, 2018.

[5] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[6] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint load balancing and offloading in vehicular edge computing and networks," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4377–4387, 2019.

[7] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1990.

[8] H. Guo, J. Liu, and J. Zhang, "Computation offloading for multi-access mobile edge computing in ultra-dense networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 14–19, 2018.

[9] D. Huang, P. Wang, and D. Niyato, "A dynamic offloading algorithm for mobile computing," *IEEE Transactions on Wireless Communications*, vol. 11, no. 6, pp. 1991–1995, 2012.

[10] Q. Huang, W. Yue, Y. Yang, and L. Chen, "P2gt: Fine-grained genomic data access control with privacy-preserving testing in cloud computing," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 1–1, 2021.

[11] C. Kai, H. Zhou, Y. Yi, and W. Huang, "Collaborative cloud-edge-end task offloading in mobile-edge computing networks with limited communication capability," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 2, pp. 624–634, 2021.

[12] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge computing: A survey," *Future Generation Computer Systems*, vol. 97, pp. 219–235, 2019.

[13] L. Li, K. Ota, and M. Dong, "Deepnfv: A lightweight framework for intelligent edge network functions virtualization," *IEEE Network*, vol. 33, no. 1, pp. 136–141, 2019.

[14] Z. Li, H. Chen, K. Lin, V. Shakhov, L. Shi, and J. Yu, "From edge data to recommendation: A double attention-based deformable convolutional network," *Peer-to-Peer Networking and Applications*, pp. 1–14, 2021.

[15] K. Lin, H. Chen, N. Yan, Z. Li, J. Li, and N. Jiang, "Fast and reliable missing tag detection for multiple-group rfid systems," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2021.

[16] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu, "A scalable and quick-response software defined vehicular network assisted by mobile edge computing," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 94–100, 2017.

[17] Y. Liu, S. Wang, J. Huang, and F. Yang, "A computation offloading algorithm based on game theory for vehicular edge networks," in *Proc. IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6.

[18] Y. Liu, H. Yu, S. Xie, and Y. Zhang, "Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11 158–11 168, 2019.
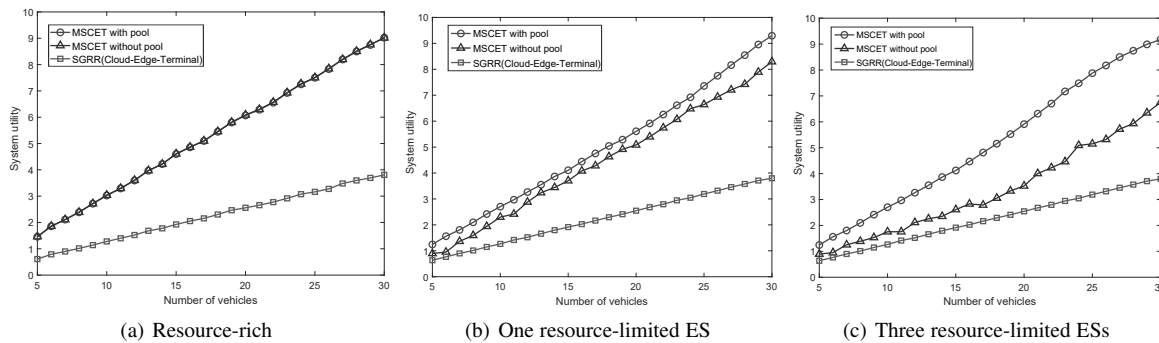
Fig. 6. Performance comparison of system utility under different amount of resource with the increase of vehicles in overlapping region.

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

[20] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. USENIX Conference on Hot Topics in Cloud Computing*, vol. 10, 2010.

[21] L. Pu, X. Chen, G. Mao, Q. Xie, and J. Xu, "Chimera: An energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 84–99, 2019.

[22] Y. Qiu, T. Misu, and C. Busso, "Analysis of the relationship between physiological signals and vehicle maneuvers during a naturalistic driving study," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 3230–3235.

[23] E. E. Seabolt, G. Nayar, H. Krishnareddy, A. Agarwal, K. L. Beck, I. Terrizzano, E. Kandogan, M. Kuntomi, M. Roth, V. Mukherjee, and J. H. Kaufman, "Ibm functional genomics platform, a cloud-based platform for studying microbial life at scale," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 1–1, 2020.

[24] B. Shen, X. Xu, F. Dai, L. Qi, X. Zhang, and W. Dou, "Dynamic task offloading with minority game for internet of vehicles in cloud-edge computing," in *2020 IEEE International Conference on Web Services (ICWS)*, 2020, pp. 372–379.

[25] Y. Song, S. S. Yau, R. Yu, X. Zhang, and G. Xue, "An approach to qos-based task distribution in edge computing networks for iot applications," in *Proc. IEEE International Conference on Edge Computing (EDGE)*, 2017, pp. 32–39.

[26] I. Sorkhoh, D. Ebrahimi, R. Atallah, and C. Assi, "Workload scheduling in vehicular networks with edge cloud capabilities," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8472–8486, 2019.

[27] Q. Wu, X. Chen, Z. Zhou, L. Chen, and J. Zhang, "Deep reinforcement learning with spatio-temporal traffic forecasting for data-driven base station sleep control," *IEEE/ACM Transactions on Networking*, vol. 29, no. 2, pp. 935–948, 2021.

[28] F. Xia, A. M. Ahmed, L. T. Yang, and Z. Luo, "Community-based event dissemination with optimal load balancing," *IEEE Transactions on Computers*, vol. 64, no. 7, pp. 1857–1869, 2015.

[29] F. Xia, A. Rahim, X. Kong, M. Wang, Y. Cai, and J. Wang, "Modeling and analysis of large-scale urban mobility for green transportation," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1469–1481, 2018.

[30] C. You, K. Huang, H. Chae, and B. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, 2017.

[31] Z. Yu, Y. Gong, S. Gong, and Y. Guo, "Joint task offloading and resource allocation in uav-enabled mobile edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3147–3159, 2020.

[32] F. Zeng, R. Zhang, X. Cheng, and L. Yang, "Channel prediction based scheduling for data dissemination in vanets," *IEEE Communications Letters*, vol. 21, no. 6, pp. 1409–1412, 2017.

[33] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task offloading in vehicular edge computing networks: A load-balancing solution," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2092–2104, 2020.

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

[35] J. Zhao, Y. Liu, K. K. Chai, Y. Chen, and M. Elkashlan, "Many-to-many matching with externalities for device-to-device communications," *IEEE Wireless Communications Letters*, vol. 6, no. 1, pp. 138–141, 2017.
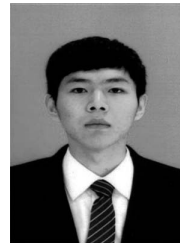
Zhichen Ni received the B.E. degree in measurement and control technology and instrumentation from China University of Petroleum, China, in 2019. He is currently pursuing his master degree in control science and engineering in the College of Control Science and Engineering, China University of Petroleum, China. His current research interests include edge computing and edge intelligence.

Honglong Chen received the M.E. degree in control science and engineering from Zhejiang University, China, in 2008, and the Ph.D degree in computer science from The Hong Kong Polytechnic University, Hong Kong, in 2012. He was a Postdoctoral Researcher in the School of CIDSE at Arizona State University from 2015 to 2016. He is currently a Professor and Ph.D supervisor with the College of Control Science and Engineering, China University of Petroleum, China. His current research interests are in the areas of Internet of Things, edge computing and crowdsensing. He has published more than 80 research papers in prestigious journals and conferences including IEEE TIFS, IEEE TMC, IEEE TWC, IEEE TVT, IEEE IoT-J, IEEE INFOCOM, IEEE ICPP, IEEE ICDCS, etc. He is a senior member of IEEE and CCF (China Computer Federation), and a member of ACM.
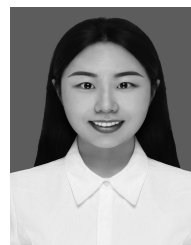
Zhe Li received the B. E. degree in automation from Shandong University of Science and Technology, China, in 2018. She is currently a graduate student in the College of Control Science and Engineering, China University of Petroleum, China. Her research interest is in the field of edge intelligence.

Xiaomeng Wang received the B.E. degree in measurement and control technology and instrumentation from China University of Petroleum, China, in 2020. She is currently pursuing her master degree in control science and engineering in the College of Control Science and Engineering, China University of Petroleum, China. Her current research interests include cybersecurity and artificial intelligence security.

Na Yan received the B. E. degree in Automation from China University of Petroleum, China, in 2019. She is currently pursuing her master degree in control science and engineering in the College of Control Science and Engineering, China University of Petroleum, China. Her research interest is in the field of RFID.

Weifeng Liu is currently a Professor with the College of Control Science and Engineering, China University of Petroleum (East China), China. He received the double B.S. degree in automation and business administration and the Ph.D. degree in pattern recognition and intelligent systems from the University of Science and Technology of China, Hefei, China, in 2002 and 2007, respectively. His current research interests include pattern recognition and machine learning. He has authored or co-authored a dozen papers in top journals and prestigious conferences including 10 ESI Highly Cited Papers and 3 ESI Hot Papers. Dr. Weifeng Liu serves as associate editor for Neural Processing Letter, co-chair for IEEE SMC technical committee on cognitive computing, and guest editor of special issue for Signal Processing, IET Computer Vision, Neurocomputing, and Remote Sensing. He also serves dozens of journals and conferences.

Feng Xia received the B.Sc. and Ph.D. degrees from Zhejiang University, Hangzhou, China. He is currently an Associate Professor and the Discipline Leader of the School of Science, Engineering and Information Technology, Federation University Australia, Ballarat, VIC, Australia. He has published two books and over 300 scientific papers in international journals and conferences. His research interests include data science, knowledge management, social computing, and systems engineering. He is a Senior Member of IEEE and ACM.