Traffic-Aware Task Offloading Based on Convergence of Communication and Sensing in Vehicular Edge Computing

Yanli Qi, Yiqing Zhou, *Senior Member, IEEE*, Ya-Feng Liu, *Senior Member, IEEE*, Ling Liu, *Member, IEEE*, and Zhengang Pan, *Senior Member, IEEE*

***Abstract*—With the explosive growth of computation-intensive and latency-sensitive vehicular applications, limited on-board computing resources can hardly satisfy these heterogeneous requirements and task offloading becomes a potential solution. However, task offloading in vehicular networks may face the dilemma of unaffordable uploading time caused by the huge amount of uploading traffic. Therefore, considering the applications which use the environmental data as their input, the sensing abilities of serving nodes (SNs) are exploited and a traffic-aware task offloading (TATO) mechanism based on convergence of communication and sensing is proposed. In the TATO mechanism, a task vehicle can adaptively upload the input data to some SNs and transmit the computation instructions to others which use the environmental data sensed by themselves as input. The objective is to minimize the overall response time (ORT) by jointly optimizing the task and wireless bandwidth ratios. Next, a binary search and feasibility check (BSFC) algorithm is designed to solve the optimization problem. Simulation results demonstrate the effectiveness of the proposed BSFC algorithm and show that the TATO mechanism always outperforms the benchmark mechanisms (i.e., communication-based offloading and sensing-based offloading) in terms of the ORT. Specifically, when the task offloading traffic is huge and the wireless transmission capability becomes a bottleneck, TATO can reduce the ORT by 42.8% compared with that of the communication-based offloading.**

***Index Terms*—Convergence of communication and sensing, overall response time (ORT), traffic-aware task offloading, vehicular network.**

Manuscript received December 31, 2020; revised April 18, 2021; accepted May 15, 2021. Date of publication May 24, 2021; date of current version December 7, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1807803; in part by the National Natural Science Youth Fund of China under Grant 61901452; and in part by Innovative Project of ICT/CAS under Grant 20196110. The work of Ya-Feng Liu was supported in part by the National Natural Science Foundation of China under Grant 12022116, Grant 12021001, Grant 11991021, and Grant 11688101. *(Corresponding author: Yiqing Zhou.)*

Yanli Qi, Yiqing Zhou, and Ling Liu are with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: qiyanli@ict.ac.cn; zhouyiqing@ict.ac.cn; liuling@ict.ac.cn).

Ya-Feng Liu is with the State Key Laboratory of Scientific and Engineering Computing, Institute of Computational Mathematics and Scientific/Engineering Computing, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China (e-mail: yafliu@lsec.cc.ac.cn).

Zhengang Pan is with the Advanced Technology Research Institute of Unisoc, Beijing 100191, China (e-mail: zhengang.pan@unisoc.com).

Digital Object Identifier 10.1109/JIOT.2021.3083065

I. Introduction

RECENTLY, Internet of Vehicles has been a hot research topic in both industries and academia [1]–[3]. It is predicted that the number of actively connected vehicles will approach to 152 million by 2020 [4]. Meanwhile, more and more sensors (e.g., cameras, LIDAR) are integrated on vehicles to sense the environmental information. They will generate approximately 1-GB data per second, which should be processed by the on-board processors in real time for safety driving [5]. Furthermore, various computation-intensive applications for vehicles, such as online gaming, virtual reality/augmented reality (VR/AR), etc., continue to proliferate [6], [7]. However, existing on-board processors cannot satisfy such high computing requirements.

One possible solution for this problem is to offload tasks from vehicles to network servers with relatively powerful computing capability [8]–[10]. Mobile edge computing (MEC) with network edge servers is a promising paradigm, which can achieve low latency due to the short distance between vehicles and edge servers [11]–[15]. Meanwhile, more and more vehicles are equipped with advanced computing processors, and vehicle-as-a-resource or vehicular cloud computing (VCC) has been proposed to exploit the under-utilized resources of parked and moving vehicles. For example, parked vehicles can be taken as static backbones providing computing resources to vehicles, pedestrians, or people in shopping malls [16]–[18]. Moreover, moving vehicles can also construct a vehicular cloud to provide computing and storage services to neighboring vehicles [19]–[21]. However, VCC cannot work alone. Due to the limited communication distance between vehicles, it is possible that there are no vehicles within the task vehicle's (TaV's) communication range to form a vehicular cloud. In this case, the TaV should work with MEC co-located with roadside units (RSUs) or base stations. Therefore, the collaboration between MEC and VCC is necessary. In this article, the system with both VCC and MEC are denoted as vehicular edge computing (VEC), and both serving vehicles and MEC servers are serving nodes (SNs).

Many recent research efforts have been dedicated to task offloading in VEC systems [22]–[27], and the task overall response time (ORT) is always concerned as an important performance metric, which usually includes the input data and output transmission time, the queuing time, and the computing time. For example, [22] aimed to minimize the VR task response time by dividing a VR task into two subtasks and processing them at the MEC server and the vehicle separately and simultaneously. Simulation results showed that the task response time can be decreased significantly by offloading the task and optimizing the resource allocation strategy reasonably. It should be noted that in [22], the input data size is about 80 MB and its transmission time accounts for about 94% of the ORT. In addition to offloading partial tasks to the MEC server and adjacent vehicles, local computing was also considered in [23]. Simulations showed that the input data transmission time accounts for about 50% of the ORT. Note that [22] and [23] assumed that the task node has the full information when making the offloading decisions, such as the computing capabilities and loads, the moving directions, and speeds of the serving nodes. However, this is impractical for the distributed vehicles.

Therefore, a more practical scenario with incomplete information was considered in [24] where the TaV cannot make the decision directly. Gu *et al.* [24] performed the task assignment in a distributed manner and used a one-to-many matching game to model the task assignment problem. Simulations showed that the input data transmission time accounts for about 73% of the ORT. Besides the distributed matching game, introducing a centralized unit (CU) to collect the full information and make offloading decisions is also feasible. Zhou *et al.* [25] utilized an unmanned aerial vehicle to collect computing tasks from devices and make online offloading decisions. The objective was to minimize the transmission and computing time of all tasks. Ye *et al.* [26] considered a cloud radio access network (C-RAN) architecture for offloading decision making and the input data transmission time contributed about 98% of the ORT. Different from previous studies [22]–[26] that mainly focused on deterministic tasks, Wang *et al.* [27] considered dynamic tasks with message flow scheduling. Assuming that the input data transmission time is normally distributed, simulation showed that it accounted for about 36% of the ORT. It can be seen that in existing works, the input data must be transmitted to SNs during the task offloading, regardless of the uploading traffic size, which ranges from several megabits to hundreds of megabits in [22]–[27]. The input data transmission time can contribute as high as 98% to the ORT.

Note that with the explosive growth of various vehicular applications, the uploading traffic may be huge, which needs many radio resources and the uploading time may be unaffordable. Take the three-dimensional (3-D) reconstruction in self-driving as an example, which can be decomposed into multiple tasks, such as image acquisition, camera calibration, camera pose estimation, depth estimation, depth map fusion, and rendering [28]. If the camera pose estimation task needs to be offloaded, only the coordinate values of the key points in the video frames should be uploaded to SNs. In this case, the traffic is light and can be directly uploaded from TaVs to SNs. However, if the depth map fusion task needs to be offloaded, the original high resolution video frames should be uploaded to SNs for depth map fusion. Since the volume of video data is large, e.g., 11.9 Gb/s for a camera with a resolution of 7680*4320 (i.e., 8K resolution), 12 bits per pixel and 30 fps, it will take about 1.2 s to upload via 5G with an uplink peak rate of 10 Gb/s. This is unaffordable for latency-sensitive applications.

Note that many vehicular applications use the environmental data as their input, such as AR or tactile Internet technology-based self-driving. These computation-intensive applications need key technologies such as 3-D reconstruction. It can be expected that a future vehicle should install many sensors, including a sufficient number of cameras and a LIDAR to see "a whole world" from all directions [29]. Therefore, serving vehicles close to the TaVs can obtain similar environmental data using their own sensors [30], [31]. Moreover, RSUs equipped with various sensors can also sense the environmental data within its coverage. So MEC servers co-located with RSUs can obtain the environmental data similar to that obtained by the TaVs. It should be noted that SNs in different locations have different perspectives. Using the TaV's coordinate contained in the computation instructions, the environmental data sensed by SNs could be preprocessed by coordinate transformation to eliminate the differences [32]. Therefore, the instructions transmission and the environmental data sensing provide a new possibility for the task offloading. When some SNs are selected to help the TaVs to complete the computation-intensive tasks, it is possible that instead of the TaVs uploading a huge volume of the input data to SNs, SNs may use the environmental data sensed by themselves. Compared with the input data, the volume of the computation instructions will be much smaller and their transmission time can be reduced significantly.

As a whole, when task offloading is needed, due to the large difference in the input data size, the transmission capability, and the coordinate transforming rates of SNs, some SNs may choose the input data transmission, while others may choose the instructions transmission and data sensing. Therefore, SNs' transmission strategies could be adaptively selected according to the objective function, the amount of traffic, the transmission capability, and so on.

Considering computation-intensive vehicular applications using the environmental data as input, this article proposes a traffic-aware task offloading (TATO) mechanism among vehicles and MEC servers. Assume that MEC servers are co-located with a CU. All vehicles and RSUs are equipped with sensors and they can sense the environmental data in real time. Vehicles periodically report their information to the CU for controlling the system centrally, such as locations, moving speeds, wireless transmission capability, computing loads, computing capabilities, and so on. When a TaV has a computation-intensive task, the CU can make the offloading decision based on the overall system information and deliver the decision to the TaV. Similar to existing works, the objective of this article is to minimize the ORT needed by a TaV offloading its task to neighboring SNs. We propose that with the TATO mechanism, a TaV could adaptively upload the input data to some SNs and transmit the computation instructions to others. If an SN receives a computation instruction, it uses its own environmental data as input. The main contributions of this work are summarized as follows.

1) To minimize the ORT, we propose a TATO mechanism based on convergence of communication and sensing in VEC systems. Different from existing mechanisms which always upload the input data to all SNs, TATO exploits the sensing capability of SNs, and adaptively transmit the input data or the computation instructions to each SN, according to the input data size, computation instructions size, wireless transmission capability, and coordinate transforming rates. The objective is to minimize the ORT by jointly optimizing the task and wireless bandwidth ratios.
2) To solve the formulated optimization problem, we propose a binary search and feasibility check (BSFC) algorithm. The basic idea of the proposed algorithm is to perform a binary search on the ORT. The key step in the binary search procedure is to check whether all subtasks can be completed with a given ORT threshold. By carefully analyzing the relationship between the ORT and the task and wireless bandwidth ratios and exploiting the special structure, the feasibility check problem can be reformulated as an equivalent optimization problem in terms of only the wireless bandwidth ratio and can be efficiently solved by a modified gradient projection method (GPM).
3) Extensive comparisons are carried out to verify the effectiveness of TATO. Denote the task offloading mechanism with the input data uploading to all SNs as communication-based offloading (CBO), and that with the instructions transmitting to all SNs and SNs sensing the environmental data by themselves as sensing-based offloading (SBO). Simulation results verify the effectiveness of the proposed BSFC algorithm and show that the proposed TATO mechanism outperforms the pure CBO and SBO mechanisms by reducing the ORT by 28% and 11.4%, respectively. When the task offloading traffic is huge and the wireless transmission capability becomes a bottleneck, TATO can reduce the ORT by 42.8% compared with that of CBO.

The reminder of this article is organized as follows. The system model is introduced in Section II and the proposed TATO mechanism is presented in Section III. Section IV solves the formulated optimization problem. Then simulation results are shown in Section V to demonstrate the effectiveness of the proposed mechanism. Finally, conclusions are drawn in the last section.

II. System Model

Consider the vehicles travelling on a unidirectional and one-lane highway. Although bidirectional roads are more realistic, the serving time of vehicles in the opposite direction is short and ignored in this preliminary work, which concentrates on the convergence of communication and sensing. The moving vehicles are equipped with computing and storage resources and various sensors, such as LIDAR, millimeter-wave radar, and high-definition cameras, to obtain the environmental information in real time.

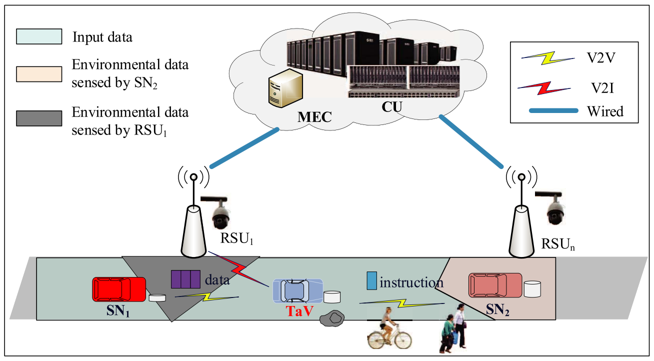

Input data
Environmental data sensed by SN_2
Environmental data sensed by RSU_1
MEC
CU
V2V
V2I
Wired
RSU_1
RSU_n
data
instruction
SN_1
TaV
SN_2

Fig. 1. Vehicular network architecture.

The vehicular network is shown in Fig. 1, which includes a CU and a couple of RSUs geographically distributed and deployed along the road. RSUs are equipped with sensors, such as high-definition cameras, and have the ability to sense the environmental information. RSUs are connected to the CU via high-speed wired fronthaul links. The CU is a global control center and is co-located with a MEC server to provide computing resources for those computation-intensive applications together with the moving vehicles. It is also possible to deploy MEC servers at the RSUs [33], [34]. However, this scheme has drawbacks, such as limited coverage of RSUs, high cost of large-scale deployment of MEC servers, and limited computing capability of MEC servers. So it is more practical to deploy MEC servers at the CU [5]. Frequency division multiplexing access techniques are employed for the TaV to communicate with serving vehicles and with the RSUs simultaneously. In addition, the uploading and feedback processes adopt time division duplexing mode.

We focus on the task offloading of one TaV in the coverage of one RSU. Consider the computation-intensive and latency-sensitive vehicular applications which use the environmental data as their input, for example, self-driving based on AR technologies. Thus technologies such as 3-D reconstruction are needed. The vehicles equipped with sensors are able to obtain the input data through real-time sensing. Similar to [22]–[27], it is assumed that the concerned task can be split arbitrarily, each subtask is end-independent and can be processed in parallel. Thus, the task can be divided into several parts for offloading.

To complete a complicated task in time, a TaV would offload some subtasks to other vehicles and/or the MEC server. The offloading decision is made by the CU based on the network status. Then the decision is sent to the TaV. Therefore, the task offloading can be implemented as shown in Fig. 2, and the details are described as follows.

Step 1: A TaV generates a complicated task and delivers the task description to the CU, such as the input data size, the output data size, the computation intensity (i.e., the required computing resources for one bit input data) [35], [36], the QoS requirements, and so on.

Step 2: After receiving the task description, the CU makes the offloading decision according to the network

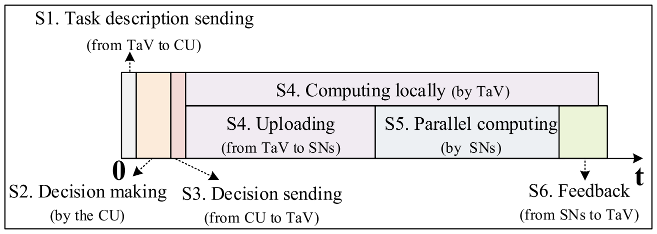


Fig. 2. Implementation of task offloading.

status, including the status of the MEC server, wireless transmission capability of RSUs, and the information reported by each vehicle, such as location, moving direction and speed, wireless transmission capability, and computing capability and load.

Step 3: The CU sends the offloading decision to the TaV.

Step 4: According to the received offloading decision, the TaV splits the task and executes the subtask allocated to itself locally. Simultaneously, the TaV uploads subtasks to the serving vehicles and the RSU via wireless links. Then the RSU relays the subtask to the MEC server via the wired fronthaul link.

Step 5: After receiving the subtasks, SNs compute them in parallel.

Step 6: Each SN feedbacks its output to the TaV.

It can be seen that in the conventional task offloading, the input data is always uploaded from a TaV to SNs in Step 4 regardless of the uploading traffic size [22]–[27]. However, with the diversification of vehicular applications and services, a huge difference has been observed in the uploading traffic. When the amount of the input data is large, uploading the input data not only consumes a lot of communication resource, but also leads to an unaffordable transmission time. Therefore, a new transmission strategy should be explored to address this problem in vehicular networks.

III. Proposed TATO Mechanism

A. TATO Mechanism

In this section, a new task offloading mechanism, i.e., TATO, is proposed. Consider applications, such as 3-D reconstruction, which use the environmental data as their input and are popular in vehicular networks. Since vehicles and RSUs are equipped with sensors and have the capability to obtain the environmental data, it is possible that neighbor serving vehicles and RSUs sense the environmental data using their own sensors. Thus in Step 4, the TaV could upload the computation instructions to SNs instead of a huge volume of the input data. In this way, the transmission time can be reduced significantly.

As a matter of fact, to ensure a whole world from all directions for a self-driving vehicle, a sufficient number of cameras are installed around the vehicle (such as the Telsa) and/or a LIDAR is mounted on the top of the vehicle (such as Baidu Apollo and Waymo). Moreover, sensors of RSUs are usually installed at a high place to monitor the environment within its coverage (with the coverage radius usually being 200 m). Specifically, the sensing ranges of cameras and LIDAR are about 80 m and 150 m, respectively. Considering the cost, this article assumes that each vehicle is equipped with two cameras installed at the front and rear ends and the sensing range of each camera is 80 m. Therefore, RSUs can be deployed along the road at a regular interval of 160 m to ensure the sensing coverage of sensors. For a TaV, its adjacent serving vehicles and the nearest RSU can obtain similar environmental data, although the perspectives are different. To eliminate the difference and be consistent with the input data obtained at the TaV, the environmental data sensed by SNs should be preprocessed first, i.e., coordinate transformation. This can be realized by matrix operations using the TaV's coordinate contained in the computation instructions [32].

Therefore, during the task offloading from the TaV to SNs, there are two possible transmission processes. The first is conventional data transmission (DataT), i.e., the input data is transmitted to an SN via a wireless link. If the SN is the MEC server, the data is transmitted to an RSU first and then relayed to the MEC server via the wired fronthaul link. The second is instruction transmission (InsT) with coordinated environmental sensing, i.e., the computation instructions are transmitted to an SN via a wireless link and the offloaded subtasks are carried out by the SN using the environmental data sensed by itself. If the SN is the MEC server, the transmission of the computation instructions from the TaV is not necessary and the RSU just sends the environmental data sensed by itself to the MEC server via wired links. This is because the offloading decision is made by the CU and relayed to the TaV via the RSU. The CU and RSU can cache the offloading decision for sensing and preprocessing the corresponding environmental data and executing the subtasks.

Hence, a new TATO mechanism is proposed as follows. When a TaV has to offload its complicated task to neighboring SNs, it adaptively chooses DataT or InsT for each SN to minimize the ORT. And when an SN receives a computation instruction, its sensed environmental data should be preprocessed first. Different from existing mechanisms which always upload the input data to all SNs, TATO is aware of the traffic and can adaptively upload the input data to some SNs and transmit the computation instructions to others. With TATO, the task offloading from Step 2 to Step 5 should be modified as follows.

Step 2: After receiving the task description, the CU makes the offloading decision according to the overall information of the vehicles, RSUs, and the MEC servers to satisfy the QoS requirements. The offloading decision includes the task ratios, wireless bandwidth ratios, and the adaptive transmission strategy (i.e., DataT or InsT) of each SN.

Step 3: The CU sends the offloading decision to the TaV. Meanwhile, the CU and RSU cache the offloading decision.

Step 4: The TaV splits its complicated task and executes the subtask allocated to itself locally. Simultaneously, according to the offloading decision, the TaV uploads the input data to some SNs and the computation instructions to others.

Step 5: When an SN receives the input data, it directly performs the computing task. Otherwise, when an SN receives a computation instruction, using the coordinate values contained in the instruction, it first preprocesses its sensed environmental data by performing coordinate transformation then starts the computing.

Compared with the traditional offloading mechanism, TATO has the potential to reduce the transmission time caused by the massive input data. However, this mechanism also brings additional overhead to the VEC systems, such as increasing the decision-making complexity, and extra computing caused by the coordinate transformation for those SNs when they receive the computation instructions and use the environmental data sensed by themselves. The overhead is ignored in this article. The important acronyms and notations of this article are summarized in Table I.

B. ORT of the TATO Mechanism

For future vehicular applications, such as AR or tactile Internet technologies-based self-driving, latency is a critical metric. Therefore, the objective of this article is to minimize the ORT needed by a TaV offloading its task to neighboring SNs. As shown in Fig. 2, the ORT is composed of the following parts: 1) transmission time of the task description T^{descrip} which depends on the task description size and the wireless and wired transmission rates; 2) offloading decision making time T^{ODM} which depends mainly on the complexity of the adopted algorithm; 3) offloading decision transmission time T^{ODT} which depends on the decision size and the transmission rates; 4) subtask's uploading time which depends on the transmission size, rates, and strategy (i.e., DataT or InsT); 5) subtask's parallel computing time which depends on the subtask's ratio, computation intensity, and computing capability of each SN; and 6) subtask's output feedback time which depends on the subtask's output size and transmission rates. Note that in practice, waiting time is possible if an SN is executing other tasks when it receives the TaV's subtask. In this article, a single task is assumed and the waiting time is ignored. Multitask and waiting time will be considered in future work.

Denote the TaV as V_0, the serving vehicles as $V_n, n \in \mathcal{N} := \{1, 2, \ldots, N\}$, and the MEC server as V_{N+1}. Targeting to minimize the ORT for the entire task, the task ratio x_n ($x_n \in [0, 1]$) allocated to each node $V_n, n = 0, 1, \ldots, N + 1$, the transmission strategy of each SN, and the wireless bandwidth ratio b_n ($b_n \in [0, 1]$) allocated to each SN should be jointly optimized. According to previous analysis, the first three time T^{descrip}, T^{ODM}, and T^{ODT} are constants. In addition, we use $TD \triangleq (S_{\text{data}}, S_{\text{instr}}, r_{\text{output}}, M, C_1, C_2, \ldots, C_K)$ to describe the task, where S_{data} (in bits) denotes the task's input data size, S_{instr} (in bits) denotes the computation instruction size, r_{output} denotes the output-to-input ratio, M (in cycles/bit) denotes the task computation intensity which is the required computing resources for one bit input data and mainly depends on the nature of applications [35], [36], $C_1, C_2, \ldots, C_K$ denote the 2-D coordinates of the environmental information, and K may be dozens [37].

TABLE I
IMPORTANT ACRONYMS AND NOTATIONS

TaV	Task vehicle
SN	Serving node
TATO	Traffic-aware task offloading
CBO	Communication-based offloading
SBO	Sensing-based offloading
ORT	Overall response time
BSFC	Binary search and feasibility check
GPM	Gradient projection method
T^{descrip}	Transmission time of the task description
T^{ODM}	Offloading decision making time
T^{ODT}	Offloading decision transmission time
N	Number of serving vehicles
$V_0/V_n/V_{N+1}$	TaV/ n-th serving vehicle/ MEC server
v_0/v_n	Moving speed of V_0/V_n
$x_0/x_n/x_{N+1}$	Task ratio allocated to $V_0/V_n/V_{N+1}$
$x_n^{\text{DataT}}/x_n^{\text{InsT}}$	Task ratio of V_n with DataT/InsT
b_n/b_{N+1}	Wireless bandwidth ratio allocated to V_n/V_{N+1}
S_{data}	Task's input data size
S_{instr}	Computation instruction size
r_{output}	Output-to-input ratio
M	Task computation intensity
M_{tra}	Computation intensity of coordinate transformation
$F_0/F_n/F_{N+1}$	CPU cycles of $V_0/V_n/V_{N+1}$
$T_0/T_n/T_{N+1}$	Subtasks' response time at $V_0/V_n/V_{N+1}$
$t_0^{\text{compt}}/t_n^{\text{compt}}/t_{N+1}^{\text{compt}}$	Computing time of $V_0/V_n/V_{N+1}$
$t_n^{\text{data}}/t_{N+1}^{\text{data}}$	Transmission time to V_n/V_{N+1} in DataT case
$t_n^{\text{instr}}/t_{N+1}^{\text{instr}}$	Transmission time to V_n/V_{N+1} in InsT case
$t_n^{\text{tra}}/t_{N+1}^{\text{tra}}$	Coordinate transformation time of V_n/V_{N+1}
$t_n^{\text{upload}}/t_{N+1}^{\text{upload}}$	Uploading time from V_0 to V_n/V_{N+1}
$t_n^{\text{output}}/t_{N+1}^{\text{output}}$	Feedback time from V_n/V_{N+1} to V_0
$R_{V_0\to V_n}/R_{V_0\to V_{N+1}}$	Wireless rate from V_0 to V_n/V_{N+1}
$R_{V_n\to V_0}/R_{V_{N+1}\to V_0}$	Wireless rate from V_n/V_{N+1} to V_0
B_{total}	Total wireless bandwidth
P_n/P_{N+1}	Uplink power allocated to V_n/V_{N+1}
P'_n/P'_{N+1}	Downlink power of V_n/V_{N+1}
$\tilde{h}_n/\tilde{h}_{N+1}$	Channel fading coefficient from V_0 to V_n/V_{N+1}
$\tilde{h}'_n/\tilde{h}'_{N+1}$	Channel fading coefficient from V_n/V_{N+1} to V_0
d_n/d_{N+1}	Distance between V_0 and V_n/V_{N+1}
p_n/p_{N+1}	Initial coordinates of V_n/V_{N+1}
$\sigma_n^2/\sigma_n'^2/\sigma_{N+1}^2/\sigma_{N+1}'^2$	Power of addictive white Gaussian noise
α	Path-loss exponent
D_0	Distance from RSU$_0$ to the centerline of road
H_0	Height of RSU$_0$
$\mathcal{N}_k^{\text{DataT}}/\mathcal{N}_k^{\text{InsT}}$	The set of serving nodes with DataT/InsT

In this article, each variable is represented by 64 bits, i.e., double float, and the task description size is a few kilobits. The offloading decision is composed by $\{x_n\}_{n=0}^{N+1}$, $\{b_n\}_1^{N+1}$, and the transmission strategies. So there are $3N + 4$ decision variables. When there are $N = 10$ serving vehicles, the offloading decision size is about 2.1 kb. It can be seen that compared with the input data, the task description and

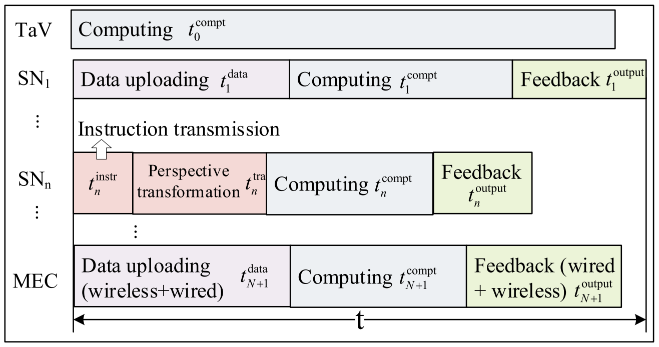

TaV
Computing t_0^{compt}
SN$_1$
Data uploading t_1^{data}
Computing t_1^{compt}
Feedback t_1^{output}
Instruction transmission
SN$_n$
t_n^{instr}
Perspective transformation t_n^{tra}
Computing t_n^{compt}
Feedback t_n^{output}
MEC
Data uploading (wireless+wired) t_{N+1}^{data}
Computing t_{N+1}^{compt}
Feedback (wired + wireless) t_{N+1}^{output}
t

Fig. 3. SRT of the proposed TATO mechanism.

offloading decision sizes are quite small and their transmission time is negligible. For the offloading decision-making time, it can be small enough by designing an optimization algorithm with lower complexity. For instance, it is about one milliseconds using BSFC with MATLAB running on a PC (Intel Core i5-7400U CPU @3.00 GHz). By optimizing the algorithm such as the parallel execution and using a high-performance server, the offloading decision-making time can be further reduced. Therefore, the objective can be simplified to minimize the parallel subtasks' response time (SRT), which includes the input data/instruction uploading, subtask computing, and output feedback time, according to steps 4-6 of the TATO mechanism. Since the subtasks are executed in parallel, their response time can be described as shown in Fig. 3. Next, the SRT at the TaV, the serving vehicles, and the MEC server will be analyzed respectively.

1) SRT at the TaV: When a subtask is executed at the TaV V_0, the uploading and feedback time is zero. And the SRT T_0 is equal to the computing time. Assume that the CPU cycles of V_0 is F_0 which remains static during the subtask's computing process. Then, the SRT is given by [35], [36]

$$T_0(x_0) = t_0^{\text{compt}}(x_0) = \frac{x_0 S_{\text{data}} M}{F_0}. \tag{1}$$

2) SRT at the Serving Vehicles: When a subtask is executed at a serving vehicle V_n, the input data or computation instruction must be transmitted to this vehicle via the wireless link first. Then V_n executes this subtask with the transmitted input data or the coordinate transformed sensed data. Finally, the output is feedback to the TaV.

Uploading model: First, the time-varying wireless uploading rate $R_{V_0 \to V_n}(t)$ from V_0 to V_n is given by

$$R_{V_0 \to V_n}(t) = b_n B_{\text{total}} \log_2\left(1 + \frac{P_n \left|\tilde{h}_n\right|^2 d_n^{-\alpha}(t)}{\sigma_n^2}\right) \tag{2}$$

where B_{total} is the total wireless bandwidth of the system, $b_n \in [0, 1]$ is the wireless bandwidth ratio allocated to V_n, P_n and σ_n^2 are the uplink transmission power and the noise power, respectively, $\tilde{h}_n$ is the complex channel fading coefficient from V_0 to V_n, $d_n^{-\alpha}(t)$ is the path loss and α is the path loss exponent [38], $d_n(t)$ denotes the time-varying distance between V_0 and V_n during the task offloading, because both vehicles are moving. Considering a unidirectional and one-lane road, the initial coordinates of V_0 and V_n are assumed to be zero and p_n, respectively, and the moving speeds of V_0 and V_n are v_0 and v_n, respectively. In particular, p_n cannot be equal to zero since only one vehicle is allowed in the same location. Then, $d_n(t)$ is given by

$$d_n(t) = \sqrt{|p_n + (v_n - v_0)t|^2}, \ p_n \neq 0, \ n \in \mathcal{N}. \tag{3}$$

Note that this article focuses on latency-sensitive applications, whose ORT is about milliseconds (such as less than 3 ms for self-driving [30]), and the speed of vehicles travelling on the highway is about 80–140 km/h. Since the TaV and serving vehicles are in the same direction, their relative speed is not high (such as 60 km/h when taking the maximum speed 140 km/h and the minimum speed 80 km/h as examples). Taking 20 ms as an example, the relative moving distance is about 0.34 m during task offloading and it can be ignored compared with $|p_n|$ since the smallest possible $|p_n|$ is a few meters to tens of meters (due to the vehicle length and safety distance). Therefore, the distance between V_0 and V_n is assumed to be fixed and given by

$$d_n = |p_n| \neq 0, \ n \in \mathcal{N}. \tag{4}$$

Then, for the DataT case, the uploading time to the serving vehicle V_n is given by

$$t_n^{\text{data}}(x_n, b_n) = \frac{x_n S_{\text{data}}}{R_{V_0 \to V_n}} = \frac{x_n S_{\text{data}}}{b_n B_{\text{total}} \log_2\left(1 + \frac{P_n \left|\tilde{h}_n\right|^2 |p_n|^{-\alpha}}{\sigma_n^2}\right)} \tag{5}$$

where $x_n S_{\text{data}}$ denotes the amount of the input data transmitted to V_n. Similarly, for the InsT case, the transmission time of a computation instruction is given by

$$t_n^{\text{instr}}(b_n) = \frac{S_{\text{instr}}}{R_{V_0 \to V_n}} = \frac{S_{\text{instr}}}{b_n B_{\text{total}} \log_2\left(1 + \frac{P_n \left|\tilde{h}_n\right|^2 |p_n|^{-\alpha}}{\sigma_n^2}\right)}. \tag{6}$$

In fact, the computation instruction only contains several coordinates of the environmental information and the average number may vary from several to dozens [37]. Therefore, the computation instruction size is about hundreds to thousands of bits and its transmission time may not affect the optimized solution. Hence, the transmission time of a computation instruction t_n^{instr} is ignored in analysis and its impact together with T^{descrip}, T^{ODM}, and T^{ODT} will be investigated in simulations. As a result, the uploading time of serving vehicles for the InsT case is mainly decided by the coordinate transformation time. Assuming that the serving vehicles sense the environmental data all the time, the coordinate transformation can be implemented immediately after receiving the computation instruction. The transformation time depends on the amount of sensed data, given by

$$t_n^{\text{tra}}(x_n) = \frac{x_n S_{\text{data}} M_{\text{tra}}}{F_n} \tag{7}$$

where M_{tra} denotes the computation intensity of the coordinate transformation and F_n denotes the CPU cycles of V_n, which remains static during the subtask's computing process.

Using TATO, there is design freedom of choosing the transmission strategy of each V_n, i.e., DataT or InsT. To minimize the SRT of V_n, it is desirable to choose the smaller uploading time with DataT and InsT. The uploading time from V_0 to V_n can thus be compactly written as

$$t_n^{\text{upload}}(x_n, b_n) = \min\left\{t_n^{\text{data}}(x_n, b_n), t_n^{\text{tra}}(x_n)\right\}. \tag{8}$$

Computing Model: After uploading, the serving vehicles execute the allocated subtasks in parallel and the computing time is given by

$$t_n^{\text{compt}}(x_n) = \frac{x_n S_{\text{data}} M}{F_n}. \tag{9}$$

Feedback Model: Finally, the serving vehicles feedback the computing outputs to the TaV. Assume that the allocated wireless bandwidth for the output feedback remains the same as the uploading process, the serving vehicle's transmission power is P_n', the noise power is $\sigma_n'^2$, and the channel fading coefficient from V_n to V_0 is $\tilde{h}_n'$. The wireless transmission rate from V_n to V_0 is also assumed to be fixed during the feedback and given by

$$R_{V_n \to V_0} = b_n B_{\text{total}} \log_2\left(1 + \frac{P_n'\left|\tilde{h}_n'\right|^2 d_n^{-\alpha}}{\sigma_n'^2}\right). \tag{10}$$

The feedback time is given by

$$\begin{aligned} t_n^{\text{output}}(x_n, b_n) &= \frac{x_n S_{\text{data}} r_{\text{output}}}{R_{V_n \to V_0}} \\ &= \frac{x_n S_{\text{data}} r_{\text{output}}}{b_n B_{\text{total}} \log_2\left(1 + \frac{P_n'\left|\tilde{h}_n'\right|^2 |p_n|^{-\alpha}}{\sigma_n'^2}\right)}. \end{aligned} \tag{11}$$

As a whole, the SRT at a serving vehicle is given by

$$\begin{aligned} T_n(x_n, b_n) = {} & t_n^{\text{upload}}(x_n, b_n) + t_n^{\text{compt}}(x_n) \\ & + t_n^{\text{output}}(x_n, b_n), \ n \in \mathcal{N}. \end{aligned} \tag{12}$$

3) SRT at the MEC Server: When a subtask is executed at the MEC server V_{N+1}, its SRT contains the uploading time (wireless and wired transmission time), the computing time, and the feedback time.

Uploading Model: Assume that the wireless bandwidth ratio allocated to RSU$_0$ in the uploading process is b_{N+1}, the uplink transmission power is P_{N+1}, and the noise power is σ_{N+1}^2. The channel fading coefficient from V_0 to RSU$_0$ is $\tilde{h}_{N+1}$. The time-varying wireless uploading rate $R_{V_0 \to V_{N+1}}(t)$ from V_0 to RSU$_0$ is given by

$$R_{V_0 \to V_{N+1}}(t) = b_{N+1} B_{\text{total}} \log_2\left(1 + \frac{P_{N+1}\left|\tilde{h}_{N+1}\right|^2 d_{N+1}^{-\alpha}(t)}{\sigma_{N+1}^2}\right) \tag{13}$$

where $d_{N+1}(t)$ is the time-varying distance between V_0 and RSU$_0$, which changes with the offloading time and the initial location and the moving speed of the TaV. It is given by

$$d_{N+1}(t) = \sqrt{|p_{N+1} - v_0 t|^2 + D_0^2 + H_0^2} \tag{14}$$

where p_{N+1} is the coordinates of RSU$_0$, D_0 is the distance from RSU$_0$ to the centerline of road, and H_0 is the height of RSU$_0$. In particular, taking 20 ms and the maximum speed 140 km/h as an example, the TaV only moved 0.78 m during the task offloading. Compared with D_0 (closes to the road width, such as 3 m) and H_0 (tens to hundreds of meters) [33], the moved distance has little impacts on d_{N+1}. Therefore, d_{N+1} can be taken as a constant during the task offloading and given by

$$d_{N+1} = \sqrt{|p_{N+1}|^2 + D_0^2 + H_0^2}. \tag{15}$$

And the wireless uploading rate $R_{V_0 \to V_{N+1}}$ is given by (16), shown at the bottom of the page.

First, for the DataT case, the uploading time to the MEC server is given by

$$t_{N+1}^{\text{data}}(x_{N+1}, b_{n+1}) = \frac{x_{N+1} S_{\text{data}}}{R_{V_0 \to V_{N+1}}} + \frac{x_{N+1} S_{\text{data}}}{R_{\text{wired}}} \tag{17}$$

where $x_{N+1} S_{\text{data}}$ denotes the amount of the input data transmitted to the MEC server and R_{wired} represents the wired transmission rate. In (17), the first term is the wireless transmission time from V_0 to RSU$_0$, and the second term is the wired transmission time from RSU$_0$ to the MEC server.

Next, the uploading time for the InsT case is analyzed. Since the offloading decision is cached in RSU$_0$ and the MEC server, only the sensed environmental data need to be transmitted from RSU$_0$ to the MEC server via the wired link. The transmission time is given by

$$t_{N+1}^{\text{instr}}(x_{N+1}) = \frac{x_{N+1} S_{\text{data}}}{R_{\text{wired}}}. \tag{18}$$

Then the MEC server implements coordinate transformation, and this time is given by

$$t_{N+1}^{\text{tra}}(x_{N+1}) = \frac{x_{N+1} S_{\text{data}} M_{\text{tra}}}{F_{N+1}} \tag{19}$$

where F_{N+1} denotes the CPU cycles of the MEC server, which remains static during the subtask's computing process. Thus, when RSU$_0$ needs to sense the environmental data and sends it to the MEC server, the uploading time needed is $t_{N+1}^{\text{instr}}(x_{N+1}) + t_{N+1}^{\text{tra}}(x_{N+1})$.

$$R_{V_0 \to V_{N+1}} = b_{N+1} B_{\text{total}} \log_2\left(1 + \frac{P_{N+1}\left|\tilde{h}_{N+1}\right|^2 \left(|p_{N+1}|^2 + D_0^2 + H_0^2\right)^{-\alpha/2}}{\sigma_{N+1}^2}\right) \tag{16}$$

Similar to the serving vehicles, with TATO, the uploading time from V_0 to the MEC server V_{N+1} is defined as the smaller uploading time with DataT and InsT and given by (20), shown at the bottom of this page.

Computing Model: Given the input data, the MEC server can execute the allocated subtask and the computing time is given by

$$t_{N+1}^{\text{compt}}(x_{N+1}) = \frac{x_{N+1}S_{\text{data}}M}{F_{N+1}}. \tag{21}$$

Feedback Model: Next, the MEC server feedbacks its computing output to the TaV. Assume the allocated wireless bandwidth for the output feedback remains the same as the uploading process, the transmission power of RSU$_0$ is P'_{N+1}, the noise power is σ'^2_{N+1}, the channel fading coefficient is $\tilde{h}'_{N+1}$. The wireless transmission rate from RSU$_0$ to V_0 is given by (22), shown at the bottom of the page. And the feedback time is given by

$$t_{N+1}^{\text{output}}(x_{N+1}, b_{N+1}) = \frac{x_{N+1}S_{\text{data}}r_{\text{output}}}{R_{\text{wired}}} + \frac{x_{N+1}S_{\text{data}}r_{\text{output}}}{R_{V_{N+1}\to V_0}}. \tag{23}$$

In summary, the SRT at the MEC server is given by (24), shown at the bottom of the page.

C. Problem Formulation

Since the subtasks are executed in parallel, the objective is to minimize the longest SRT among all TaVs and the MCE server. The optimization problem is formulated as follows:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{b}} \quad & T(\mathbf{x}, \mathbf{b}) := \max_{n=0,\ldots,N+1} \{T_n(x_n, b_n)\} \\ \text{s.t.} \quad & \text{C1}: \sum_{n=0}^{N+1} x_n = 1 \\ & \text{C2}: \sum_{n=1}^{N+1} b_n \le 1 \\ & \text{C3}: 0 \le x_n \le 1, \ n = 0, 1, \ldots, N+1 \\ & \text{C4}: b_n \ge 0, \ n = 1, 2, \ldots, N+1 \end{aligned} \tag{P1}$$

where constraint C1 guarantees that the sum of task ratios allocated to all nodes equals one (i.e., all subtasks are allocated), constraint C2 guarantees that the sum of wireless bandwidth ratios allocated to all SNs cannot exceed one, and constraints C3 and C4 guarantee that the task and wireless bandwidth ratios allocated to each node are nonnegative and cannot exceed one.

IV. Proposed Algorithm

Since the design variables x_n and b_n are coupled together and there is an max operator in the objective function, problem P1 is a nonconvex nonsmooth problem. Moreover, the problem is of (hidden) combinatorial nature, as each node needs to make a choice of uploading the data or the instruction. Therefore, problem P1 is generally difficult to solve. In this section, we propose an efficient algorithm for solving problem P1 by carefully exploiting its special structure. The basic idea of the proposed algorithm is to perform a binary search on the ORT. The key in the binary search algorithm is to check whether there exists a feasible pair of **x** and **b** such that all subtasks can be completed within a given ORT threshold. Fortunately, the above feasibility check problem can be reformulated as an optimization problem in terms of only **b** and solved efficiently.

A. Binary Search Over the ORT

To use the binary search over the ORT $T(\mathbf{x}, \mathbf{b})$ in problem P1, for each $n = 0, 1, \ldots, N+1$, the relationship between the SRT $T_n(x_n, b_n)$ and the task and bandwidth ratios x_n and b_n is first simplified and analyzed.

For the TaV, the SRT $T_0(x_0)$ depends only on the task ratio x_0 according to (1). For the serving vehicle $n = 1, 2, \ldots, N$, the SRT $T_n(x_n, b_n)$ is given by

$$T_n(x_n, b_n) = \min\left\{c_{n,1}\frac{x_n}{b_n},\ c_{n,2}x_n\right\} + c_{n,3}x_n + c_{n,4}\frac{x_n}{b_n} \tag{25}$$

$$t_{N+1}^{\text{upload}}(x_{N+1}, b_{N+1}) = \min\left\{t_{N+1}^{\text{data}}(x_{N+1}, b_{N+1}), t_{N+1}^{\text{instr}}(x_{N+1}) + t_{N+1}^{\text{tra}}(x_{N+1})\right\} \tag{20}$$

$$R_{V_{N+1}\to V_0} = b_{N+1}B_{\text{total}}\log_2\left(1 + \frac{P'_{N+1}\left|\tilde{h}'_{N+1}\right|^2\left(|p_{N+1}|^2 + D_0^2 + H_0^2\right)^{-\alpha/2}}{\sigma'^2_{N+1}}\right) \tag{22}$$

$$T_{N+1}(x_{N+1}, b_{N+1}) = t_{N+1}^{\text{upload}}(x_{N+1}, b_{N+1}) + t_{N+1}^{\text{compt}}(x_{N+1}) + t_{N+1}^{\text{output}}(x_{N+1}, b_{N+1}) \tag{24}$$

$$T_{N+1}(x_{N+1}, b_{N+1}) = \min\left\{c_{N+1,1}\frac{x_{N+1}}{b_{N+1}} + c_{N+1,2}x_{N+1}, \left(c_{N+1,2} + c_{N+1,3}\right)x_{N+1}\right\} + \left(c_{N+1,4} + c_{N+1,5}\right)x_{N+1} + c_{N+1,6}\frac{x_{N+1}}{b_{N+1}} \tag{26}$$

where

$$c_{n,1} = \frac{S_{\text{data}}}{B_{\text{total}}\log_2\left(1 + \frac{P_n\left|\tilde{h}_n\right|^2|p_n|^{-\alpha}}{\sigma_n^2}\right)}, \; c_{n,2} = \frac{S_{\text{data}}M_{\text{tra}}}{F_n}$$

$$c_{n,3} = \frac{S_{\text{data}}M}{F_n}, \text{ and } c_{n,4} = \frac{S_{\text{data}}r_{\text{output}}}{B_{\text{total}}\log_2\left(1 + \frac{P'_n\left|\tilde{h}'_n\right|^2|p_n|^{-\alpha}}{\sigma_n'^2}\right)}$$

are all constants. For the MEC server, the SRT $T_{N+1}(x_{N+1}, b_{N+1})$ is given by (26), shown at the bottom of the previous page, where

$$c_{N+1,1} = \frac{S_{\text{data}}}{B_{\text{total}}\log_2\left(1 + \frac{P_{N+1}\left|\tilde{h}_{N+1}\right|^2\left(|p_{N+1}|^2+D_0^2+H_0^2\right)^{-\alpha/2}}{\sigma_{N+1}^2}\right)}$$

$$c_{N+1,2} = \frac{S_{\text{data}}}{R_{\text{wired}}}, \; c_{N+1,3} = \frac{S_{\text{data}}M_{\text{tra}}}{F_{N+1}}, \; c_{N+1,4} = \frac{S_{\text{data}}M}{F_{N+1}}$$

$$c_{N+1,5} = \frac{S_{\text{data}}r_{\text{output}}}{R_{\text{wired}}}, \text{ and}$$

$$c_{N+1,6} = \frac{S_{\text{data}}r_{\text{output}}}{B_{\text{total}}\log_2\left(1 + \frac{P'_{N+1}\left|\tilde{h}'_{N+1}\right|^2\left(|p_{N+1}|^2+D_0^2+H_0^2\right)^{-\alpha/2}}{\sigma_{N+1}'^2}\right)}$$

are all constants.

According to (25) and (26), the SRT of each node only depends on its own allocated task and wireless bandwidth ratios, which is a very crucial problem structure that will be exploited in the algorithmic design. In particular, given any pair of the task ratio and the wireless bandwidth ratio, each subtask's uploading time can be computed, which is the smaller one between DataT and InsT. Then, the transmission strategy can be determined and the task's ORT can further be obtained. Moreover, the SRT of each node is a monotonically increasing function (and in particular a linear function) of its task ratio (given the wireless bandwidth ratio) and is a decreasing function of its wireless bandwidth ratio (given the task ratio).

Next, we present the binary search procedure for solving problem P1. Assume that T^* is the optimal objective value of problem P1. With a lower bound T_{lower} and an upper bound T_{upper} of T^* in hand, the binary search over the optimal ORT can be described as follows: set $T = (T_{\text{lower}} + T_{\text{upper}})/2$ and check whether there exists a feasible pair of $\mathbf{x}$ and $\mathbf{b}$ such that the objective value of problem P1 is less than or equal to T (i.e., whether the entire task can be completed within T). If the answer is "yes," then set T_{upper} to be T; otherwise set T_{lower} to be T. The above procedure is repeated until $T_{\text{upper}} - T_{\text{lower}} \leq \epsilon$ is satisfied, where $\epsilon > 0$ is a positive tolerance.

The computational complexity of the above binary search procedure depends on the preselected tolerance $\epsilon > 0$ as well as (the quality of) the initial lower and upper bounds. In particular, the worse-case complexity of the binary search procedure is in the order of $\log_2(1/\epsilon)$. As for the lower and upper bounds of T^*, a possible choice is to set T_{lower} to be zero (as T^* is obviously greater than zero) and to set T_{upper} to be the objective value of problem P1 at the feasible point of uniform allocation, i.e., $x_n = 1/(N+2), \; n = 0, 1, \ldots, N+1$, and $b_n = 1/(N+1), \; n = 1, 2, \ldots, N+1$.

B. Feasibility Check

The key step in the binary search procedure is to check whether there exists a feasible pair of $\mathbf{x}$ and $\mathbf{b}$ such that the objective function $T(\mathbf{x}, \mathbf{b})$ of problem P1 is less than or equal to a given $T > 0$. In this section, we focus on this feasibility check problem and develop an efficient algorithm for solving it. Our idea of solving the feasibility check problem is to transform it into an equivalent optimization problem only in terms of the bandwidth ratio $\mathbf{b}$ and modify the GPM algorithm to solve the equivalent optimization problem by judiciously taking care of the problem structure.

To derive the equivalent optimization problem, we first analyze the relationship between x_n and b_n in order for node n to complete the subtask with time $T > 0$. More specifically, given any $T > 0$, according to (1), the unique solution x_0 to the equation $T_0(x_0) = T$ is

$$x_0(T) = \frac{TF_0}{S_{\text{data}}M}. \tag{27}$$

Moreover, given $T > 0$ and b_n, since $T_n(x_n, b_n)$ is a linear function with respect to x_n according to (25) and (26), the unique solution x_n to the equation $T_n(x_n, b_n) = T$ is

$$\max\left\{x_n^{\text{DataT}}(b_n; T), x_n^{\text{InsT}}(b_n; T)\right\} \tag{28}$$

where

$$x_n^{\text{DataT}}(b_n; T) = \frac{T}{\frac{c_{n,1}}{b_n} + c_{n,3} + \frac{c_{n,4}}{b_n}}$$

and $x_n^{\text{InsT}}(b_n; T) = (T/[c_{n,2} + c_{n,3} + (c_{n,4}/b_n)])$ denote the task ratio of V_n with DataT and InsT, respectively. In a similar fashion, for the MEC server, the unique solution to the equation $T_{N+1}(x_{N+1}, b_{N+1}) = T$ is given by

$$\max\left\{x_{N+1}^{\text{DataT}}(b_{N+1}; T), x_n^{\text{InsT}}(b_{N+1}; T)\right\}$$
$$\begin{cases} x_{N+1}^{\text{DataT}}(b_{N+1}; T) = \frac{T}{\frac{c_{N+1,1}}{b_{N+1}} + c_{N+1,2} + c_{N+1,4} + c_{N+1,5} + \frac{c_{N+1,6}}{b_{N+1}}} \\ x_{N+1}^{\text{InsT}}(b_{N+1}; T) = \frac{T}{c_{N+1,2} + c_{N+1,3} + c_{N+1,4} + c_{N+1,5} + \frac{c_{N+1,6}}{b_{N+1}}} \end{cases} \tag{29}$$

where $x_{N+1}^{\text{DataT}}(b_{N+1}; T)$ and $x_{N+1}^{\text{InsT}}(b_{N+1}; T)$ denote the task ratio of the MEC server with DataT and InsT, respectively.

Now, we are ready to introduce the total task ratio maximization problem (with fixed $T > 0$) as problem P2

$$\max_{\mathbf{b}} \; f(\mathbf{b}; T) := x_0(T) + \sum_{n=1}^{N+1} \max\left\{x_n^{\text{DataT}}(b_n; T), x_n^{\text{InsT}}(b_n; T)\right\}$$
$$\text{s.t.} \; \sum_{n=1}^{N+1} b_n \leq 1$$
$$b_n \geq 0, \; n = 1, 2, \ldots, N+1 \tag{P2}$$

The connection between problems P1 and P2 is as follows: for any given $T > 0$, there exists a feasible pair of $\mathbf{x}$ and $\mathbf{b}$ such that the objective function of problem P1 is less than or equal to T if and only if there exists a feasible solution $\mathbf{b}$ such that the objective function of problem P2 is greater than or equal to 1. Therefore, in order to check the feasibility of problem P1 with given $T > 0$, it is sufficient to solve problem P2 with the same T.

In the rest part of this section, we focus on solving problem P2. In particular, we propose to apply a modified GPM to solve problem P2. At each iteration of the (classical) GPM, the algorithm computes the gradient of the objective function of problem P2, takes an ascent step along the gradient direction, and projects the point onto the feasible set. A technical issue of applying the GPM to solve problem P2 is that there are max operators in the objective function of problem P2 and the objective function is thus not smooth with respect to $\mathbf{b}$, which makes it problematic to directly compute the gradient. To overcome this technical issue, we propose to maximize a lower bound of the objective function of problem P2 instead of the objective function itself [39].

More specifically, suppose the current point is $\mathbf{b}_k$, where k is the iteration index. Based on the current point, we can determine the transmission strategy with DataT and InsT. Let $\mathcal{N}_k^{\text{DataT}}$ denote the set of SNs who choose to upload the input data and $\mathcal{N}_k^{\text{InsT}}$ denote the set of SNs who use the sensed environmental data corresponding to the current point $\mathbf{b}_k$. Then we apply the GPM to solve the surrogate problem P3

$$\begin{aligned}\max_{\mathbf{b}} \ & f_k(\mathbf{b}; T) := x_0(T) + \sum_{n\in\mathcal{N}_k^{\text{DataT}}} x_n^{\text{DataT}}(b_n; T) \\ & \qquad + \sum_{n\in\mathcal{N}_k^{\text{InsT}}} x_n^{\text{InsT}}(b_n; T) \\ \text{s.t.} \ & \sum_{n=1}^{N+1} b_n \le 1 \\ & b_n \ge 0, \ n = 1, \dots, N+1. \end{aligned} \tag{P3}$$

and let the returned solution be $\mathbf{b}_{k+1}$. There are two possible cases: case (I) where the correponding transmission strategy (i.e., $\mathcal{N}_{k+1}^{\text{DataT}}$ and $\mathcal{N}_{k+1}^{\text{InsT}}$) based on $\mathbf{b}_{k+1}$ is different from the current one and case (II) the correponding transmission strategy based on $\mathbf{b}_{k+1}$ is the same as the current one. When case (I) happens, we have

$$f(\mathbf{b}_{k+1}; T) \ge f_k(\mathbf{b}_{k+1}; T) \ge f_k(\mathbf{b}_k; T) = f(\mathbf{b}_k; T) \tag{30}$$

which shows that the objective function values of problem P2 are increasing and the sequence of the objective function values will converge (due to the upper bound of problem P2). In (30), the first inequality is because that $f_k(\mathbf{b}; T)$ is a lower bound of $f(\mathbf{b}; T)$, the second inequality is due to the GPM which improves the objective values of problem P3 (after each innner gradient ascent iteration), and the equality is due to the definition of the function $f_k(\mathbf{b}; T)$. When case (II) happens, the point returned by the GPM is a stationary point of problem P3, which is also a stationary point of problem P2, because the transmission stragery of the final returned solution by the GPM is the same to that of $\mathbf{b}_k$.

Algorithm 1 Proposed BSFC Algorithm for Solving Problem P1

```
Initialization: Let $\mathbf{x}_0$ and $\mathbf{b}_0$ denote the uniform allocation ratio; set $T_{\text{lower}} = 0$ and $T_{\text{upper}} = T(\mathbf{x}_0, \mathbf{b}_0)$; set $\epsilon = 10^{-4}$ and $k = 0$;
1: while $T_{\text{upper}} - T_{\text{lower}} \ge \epsilon$ do
2:     Set $T = (T_{\text{upper}} + T_{\text{lower}})/2$;
3:     while TRUE do
4:         Apply the GPM algorithm to solve problem P3 and denote the returned solution as $\mathbf{b}_{k+1}$;
5:         Compute $\mathcal{N}_{k+1}^{\text{DataT}}$ and $\mathcal{N}_{k+1}^{\text{InsT}}$ based on $\mathbf{b}_{k+1}$;
6:         if $f(\mathbf{b}_{k+1}; T) \ge 1$ then
7:             Set $T_{\text{upper}} = T$;
8:         else
9:             if $\mathcal{N}_{k+1}^{\text{DataT}}$ is the same as $\mathcal{N}_k^{\text{DataT}}$ then
10:                Set $T_{\text{lower}} = T$;
11:            end if
12:        end if
13:        Set $k = k + 1$;
14:    end while
15: end while
```

Finally, some remarks on using the GPM to solve problem P3 are as follows. First, since the objective function in problem P3 is smooth, there is no technical issue in computing its gradient. The gradient of the objective function in problem P3 can be computed as follows: for a serving vehicle $n = 1, 2, \dots, N$

$$\frac{\partial f_k(\mathbf{b}; T)}{\partial b_n} = \begin{cases} \frac{(c_{n,1}+c_{n,4})T}{(c_{n,1}+c_{n,3}b_n+c_{n,4})^2}, & \text{if } V_n \in \mathcal{N}_k^{\text{DataT}} \\ \frac{c_{n,4}T}{(c_{n,2}b_n+c_{n,3}b_n+c_{n,4})^2}, & \text{if } V_n \in \mathcal{N}_k^{\text{InsT}} \end{cases} \tag{31}$$

and for the MEC server, the gradient is given in (32), shown at the bottom of the page. Second, the step size is selected using the Armijo line search [40]. Third, the feasible set of problem P3 is a simplex and the projection onto the simplex can be computed efficiently [41]. Finally, the proposed BSFC algorithm, which judiciously combines binary search and feasibility check steps, for solving problem P1 is summarized as Algorithm 1.

V. Performance Evaluation

In this section, simulations are carried out to verify the effectiveness of the proposed TATO mechanism. Consider the unidirectional and one-lane road along which a single RSU is deployed with the coverage of 160 m and only one TaV within its coverage. For the driving safety reason, we assume that there are up to 4 serving vehicles randomly located within the RSU's coverage. The input data size S_{data} is 25 Mb (one

$$\frac{\partial f_k(\mathbf{b}; T)}{\partial b_{N+1}} = \begin{cases} \frac{(c_{N+1,1}+c_{N+1,6})T}{(c_{N+1,1}+(c_{N+1,2}+c_{N+1,4}+c_{N+1,5})b_{N+1}+c_{N+1,6})^2}, & \text{if } V_{N+1} \in \mathcal{N}_k^{\text{DataT}} \\ \frac{c_{N+1,6}T}{((c_{N+1,2}+c_{N+1,3}+c_{N+1,4}+c_{N+1,5})b_{N+1}+c_{N+1,6})^2}, & \text{if } V_{N+1} \in \mathcal{N}_k^{\text{InsT}}. \end{cases} \tag{32}$$

TABLE II
SYSTEM PARAMETERS

Parameters	Values
Transmission power P_n and P_{N+1}	$\frac{P_{\text{TaV}}}{N+1}$ W, P_{TaV}=1 W [44]
Transmission power P'_n	1 W [44]
Transmission power of RSU P'_{N+1}	2 W [44]
Power of noise σ_n^2, $\sigma_n'^2$, σ_{N+1}^2, $\sigma_{N+1}'^2$	10^{-13} W [35]
Path-loss exponent α	3.4
Distance D_0	3 m
Height of RSU$_0$ H_0	20 m [33]
Tolerance of the binary search ϵ	0.1 ms

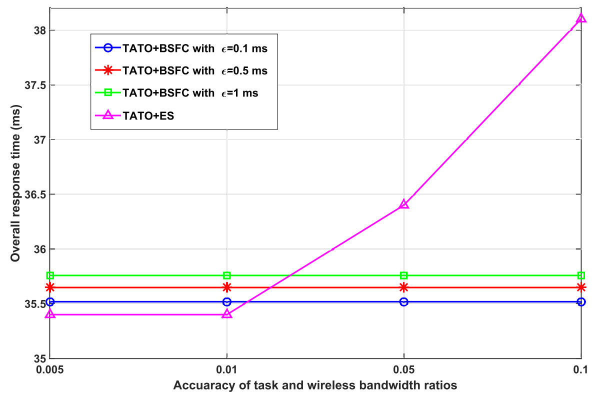


Fig. 4. Comparison of the ORT performance between BSFC and exhaustive search.

frame with a resolution of 1920*1080, 12 bits per pixel). The task computation intensity M mainly depends on the nature of applications and is set to 2640 cycles/bit [42]. Since the computing time at each node is a linear function of the computation intensity M while it is not related to the variables $\mathbf{x}$ and $\mathbf{b}$, the ORT increases linearly with M. Since our proposed TATO mechanism focuses on the uploading process, the output-to-input ratio r_{output} is initially set to 0.1 to reduce the impact of the output on the ORT. The CPU frequency of each vehicle is randomly selected within the range of $0.3 * 10^{12} - 0.6 * 10^{12}$ cycles/s and the CPU frequency of the MEC server is randomly selected within the range of $1 * 10^{12} - 2 * 10^{12}$ cycles/s [26]. The total wireless bandwidth B_{total} is set to 100 MHz which is the maximum bandwidth of the system with Sub 6 GHz in 5G NR [43]. And the wired transmission rate from the RSU to the CU is set to 100 Gb/s according to IEEE 802.3. Other system parameters used in simulations are summarized in Table II [33], [35], [44].

A. Effectiveness of TATO

In this section, the effectiveness of the proposed TATO mechanism with BSFC (TATO+BSFC) is verified. First, BSFC is compared with the exhaustive search (ES) (TATO+ES), whose computational complexity is decided by the accuracy δ of $\mathbf{x}, \mathbf{b}$ and the number of serving vehicles N, given by $O(1/\delta^{2N+3})$. The computational complexity of BSFC is $O(\log[1/\epsilon])$. Given tolerance of the binary search $\epsilon = 0.1,\ 0.5,\ 1$ ms, the ORT performance of BSFC and ES is plotted in Fig. 4 as a function of δ. It can be seen that the ORT performance of ES degrades when δ gets larger.

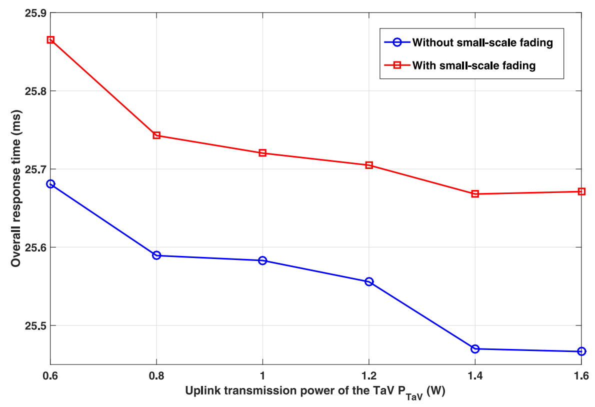


Fig. 5. ORT perfomance with/without small-scale fading.

This is because the solution returned by ES gets farther from the optimal solution due to a rougher searching with a larger δ. The ORT performance of BSFC is irrelevant with δ but degrades as ϵ increases. This is because the binary search in BSFC is terminated when the gap is less than the tolerance ϵ and the gap between the returned objective value by BSFC and the optimal objective is not larger than ϵ. When δ is small, ES can find better solutions with a slightly lower ORT than that of BSFC, at the cost of higher complexity. When δ is larger than 0.01, the ORTs of BSFC is smaller than that of ES. Overall, BSFC can achieve near optimal ORT performance with low complexity.

Note that the channel state varies rapidly due to the high-speed mobility of vehicles. Hence, previous works [44], [45] assumed that the transmission performance is mainly affected by large-scale fading and small-scale fading is ignored for the sake of simplification. Here, the impact of small-scale fading is analyzed by using TATA+BSFC where the fading coefficients $\tilde{h}_n, \tilde{h}'_n, \tilde{h}_{N+1}, \tilde{h}'_{N+1}$ are Rayleigh distributed with their entries following independently and identically distributed (independent identically distributed) $\mathcal{CN}(0, 1)$. As shown in Fig. 5, the difference in the ORTs is about 0.2 ms for the two cases with and without small-scale fading and the ORT without small-scale fading is always lower than that with small-scale fading. This is because of Jensen's inequality, i.e., $\mathbb{E}(\log_2(1 + X)) \le \log_2(1 + \mathbb{E}(X))$, where X is a nonnegative random variable and $\mathbb{E}$ denotes the expectation operator. In particular, the mean transmission rate with small-scale fading $\mathbb{E}(b_n B_{\text{total}} \log_2(1 + [(P_n |\tilde{h}_n|^2 d_n^{-\alpha})/\sigma_n^2]))$ is always less than or equal to the transmission rate without small-scale fading $b_n B_{\text{total}} \log_2(1 + [(P_n \mathbb{E}(|\tilde{h}_n|^2) d_n^{-\alpha})/\sigma_n^2])$. With a larger transmission rate, the ORT performance without small-scale fading is better. Due to the fact that the difference is small enough, the impact of small-scale fading on the ORT performance is negligible. So $|\tilde{h}_n|^2$ is set to 1 in the following simulations and the obtained ORT will be the lower bound of the case with small-scale fading.

Next, the impact of ignored times in the analysis is investigated, i.e., the task description transmission time T^{descrip}, the offloading decision making time T^{ODM}, the offloading decision transmission time T^{ODT}, and the computation instruction transmission time $t_n^{\text{instr}},\ n = 1, \ldots, N + 1$. Denote Case 1 as

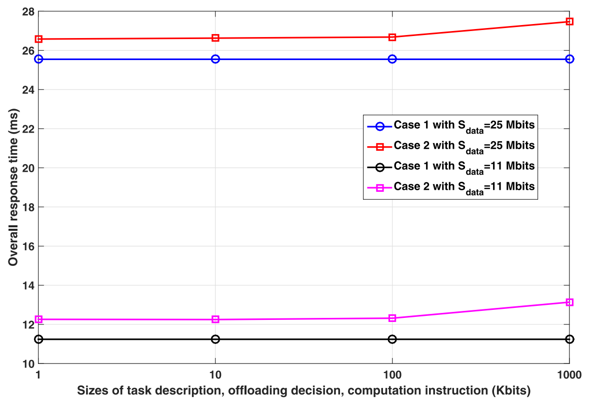


Fig. 6. ORT performance as a function of S_{instr}.

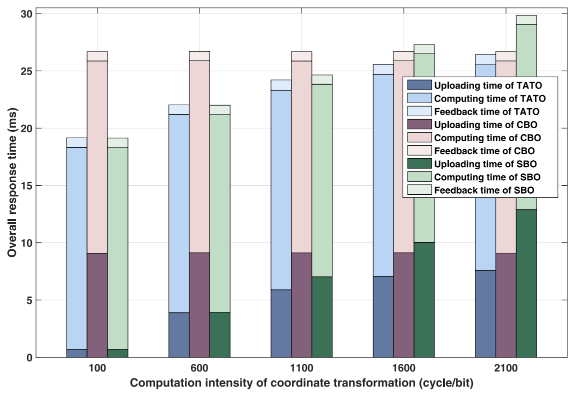


Fig. 7. ORT performance as a function of M_{tra} under different mechanisms.

TATO+BSFC considering main times without T^{descrip}, T^{ODM}, T^{ODT}, and t_n^{instr}, and Case 2 as TATO+BSFC considering all times. Without loss of generality, the sizes of task description, offloading decision, and computation instruction are set to a same value of S_{instr}. Note that T^{ODM} is irrelevant with S_{instr} and it is about 1 ms using BSFC with MATLAB running on a PC (Intel Core i5-7400U CPU @3.00 GHz). Specifically, the input data size S_{data} is set to 11 Mb (one frame with a resolution of 1280*720, 12 bits per pixel, i.e., 720P resolution) and 25 Mb (one frame with a resolution of 1920*1080, 12 bits per pixel), respectively. The ORTs of Case 1 and Case 2 are compared in Fig. 6 as a function of S_{instr}. It can be seen that the ORT of Case 1 are 11.22 ms and 25.5 ms when S_{data} is equal to 11 Mb and 25 Mb, respectively. Moreover, as S_{instr} increases, the ORT of Case 1 keeps unchanged since it is irrelevant with S_{instr} and the ORT of Case 2 remains stable first and then gradually grows. This is because, T^{descrip}, T^{ODT}, and t_n^{instr} increase linearly with S_{instr}. When S_{instr} is smaller than 100 kb, they are so small and the ORT of Case 2 is mainly decided by the times considered in Case 1 and T^{ODM}. When S_{instr} increases to 1 Mb, T^{descrip}, T^{ODT}, and t_n^{instr} increase and make a bigger contribution to the ORT, resulting in a notable increase in the ORT. In addition, when S_{instr} is smaller than 100 kb, there is a difference of about 1 ms in the ORTs between Case 1 and Case 2, which is caused by T^{ODM}. As S_{instr} increases to 1 Mb, the difference between Case 1 and Case 2 grows to about 1.9 ms. This is because, T^{descrip}, T^{ODT}, and t_n^{instr} increase the ORT by about 0.9 ms besides T^{ODM}. In practice, the description size, decision size, and computation instruction size are usually much smaller than 1 Mb. Hence the impact of T^{descrip}, T^{ODT}, and t_n^{instr} are negligible. Furthermore, T^{ODM} can also be reduced by optimizing the algorithm such as parallel execution and using a high-performance server. Therefore, it is reasonable to ignore these factors in the optimization problem.

B. Comparison of TATO, CBO, and SBO Mechanisms

In this section, two benchmark mechanisms are compared to verify the performance of our proposed TATO mechanism. The first one is the traditional offloading mechanism where the input data is always uploaded to SNs, i.e., CBO. The second one is to transmit the computation instructions to all SNs and SNs sense the environmental data by themselves, i.e., SBO.

When performing the coordinate transformation, its computation intensity M_{tra} may vary from tens cycles/bit to tens of thousands cycles/bit [46]–[48] using different algorithms. Therefore, the ORT performance of TATO is first compared with CBO and SBO in Fig. 7 as a function of M_{tra}. With the increase of M_{tra}, the ORT of TATO grows rapidly at the beginning and then gradually stabilizes, the ORT of CBO is always constant, and the ORT of SBO increases linearly. In addition, the performance of TATO always outperforms CBO and SBO. When M_{tra} is less than 1100 cycles/bit, the ORTs of TATO and SBO are similar and much smaller than that of CBO. When M_{tra} increases to 1600 cycles/bit, the ORTs of CBO and SBO are both larger than that of TATO. As M_{tra} continues to increase, the ORT of SBO gradually exceeds that of CBO and their gap gradually increases. At the same time, the gap between the ORTs of CBO and TATO gradually decreases and eventually becomes zero in our simulations. But due to the size limitations, the results after $M_{\text{tra}} = 2100$ are not shown in the figure. Taking $M_{\text{tra}} = 100$ as an example, the ratios of the uploading time of TATO and SBO are about 3.6%, while that of the CBO mechanism is about 36%. In this case, the time with InsT is smaller than the time with DataT and all SNs are selected to transmit the computation instructions. Compared with CBO, the ORT of TATO can be reduced by 28%. As M_{tra} gradually increases to 2100, the uploading time of SBO increases rapidly from 3.6% to 43%, resulting in a large ORT. However, the uploading time and the ORT of CBO keep stable since M_{tra} has no impact on CBO. As a result, the ORT of SBO gradually gets larger and finally surpasses that of CBO. Though the uploading time and the ORT of TATO also grows with M_{tra}, their growth trend gradually slows down since the system adaptively selects more SNs to transmit the input data rather than the computation instructions. Compared with SBO, the ORT of TATO can be reduced by 11.4%. When M_{tra} is large enough, the TaV transmits the input data to all SNs and the ORT of TATO equals that of CBO and no longer increases.

Next, we focus on the impact of the system parameter on the ORT, i.e., the total wireless bandwidth B_{total}. Specifically, M_{tra} is fixed to 1600 with which DataT and InsT exist at the same time in TATO. As shown in Fig. 8, the ORT decreases as B_{total} increases from 20 Mb/s. The difference is that the ORT of CBO decreases rapidly, while the ORTs of TATO and SBO

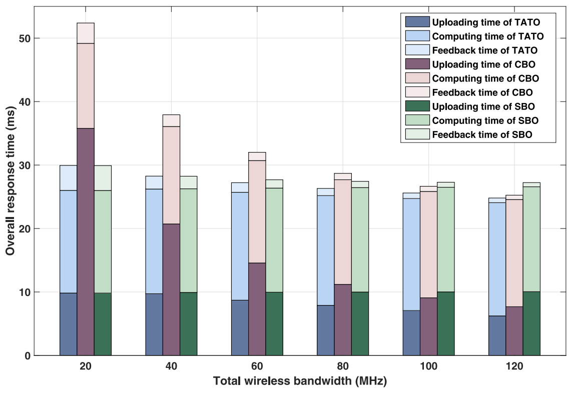


Fig. 8. ORT performance as a function of wireless bandwidth under different mechanisms.

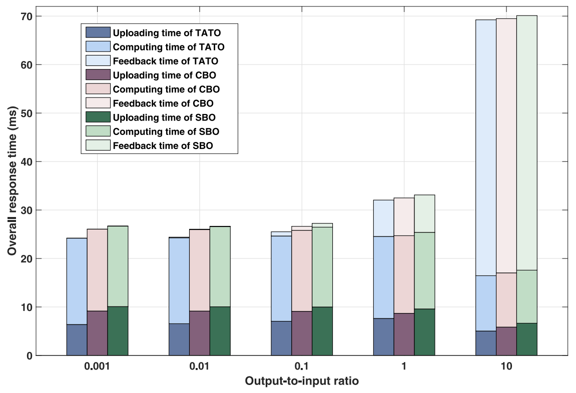


Fig. 9. ORT performance as a function of r_{output} under different mechanisms.

decrease more slowly. In addition, when B_{total} is small, the ORTs of TATO and SBO are similar and much smaller than that of CBO. TATO outperforms CBO by reducing the ORT by 42.8% when B_{total} is a bottleneck (e.g., $B_{\text{total}} = 20$ Mb/s). With the increase of B_{total}, the gap between the ORTs of TATO and CBO gradually narrows and a gap appears between the ORTs of TATO and SBO. In addition, the uploading and computing time of SBO keeps stable while the feedback time reduces obviously. This is because the instruction transmission time is ignored and B_{total} has no impact on the uploading time of SBO. However, the impact of B_{total} on the input and output data transmission is large. Thus, the change of the CBO curve is more obvious than that of the TATO and SBO curves. In TATO, all SNs select to use their sensed data to relieve the strict constraint of B_{total} at the beginning. As B_{total} increases, the system gradually selects more SNs to transmit the input data, which results in the gap between TATO and SBO. From this evaluation, the conclusion can be made that TATO always works better than CBO and SBO. In particular, when the wireless bandwidth is a bottleneck, the performance improvement of TATO will be more significant compared with the traditional task offloading mechanism CBO.

Since different vehicular applications have heterogenous requirements, the system performance will be evaluated next with different parameters related to the applications, i.e., the output-to-input ratio r_{output}. As depicted in Fig. 9, the ORTs of all mechanisms increase with r_{output}. Specifically, when r_{output} increases from 0.001 to 10, the saving ratio of the ORT decreases by TATO from 7% to 0.4% compared with CBO and decreases from 9.3% to 1.2% compared with SBO. The reason is that, the ratios of the feedback time to the ORTs increase from 0.13% to 76% for TATO, increase from 0.03% to 75.5% for CBO, and increase from 0.03% to 75% for SBO. As a result, the ratios of the uploading time gradually decrease, and the performance improvement of our proposed TATO mechanism, which aims to reduce the uploading time, degrades.

In addition, the ORT performance when varying the wired transmission rate R_{wired} from 1 Gb/s to 100 Gb/s, varying the number of serving vehicles N from 0 to 4, varying the computation intensity M from 1000 to 5000, and varying the input data size S_{data} from 0.2 Mb to 2 Gb is also compared. The TATO mechanism always outperforms the pure CBO and SBO mechanisms. In other words, our proposed mechanism is suitable for different vehicular applications with heterogeneous requirements. Limited by the number of figures, these results are not shown in this article.

The above simulations only consider the one-lane scenario. Next, the comparison of one-lane and two-lane scenarios is evaluated. It is assumed that two serving vehicles randomly locate in each lane. The lane closer to the RSU is denoted as Lane 1 and the other one is denoted as Lane 2. The scenarios where the TaV is located in Lane 1 and Lane 2 are denoted as Type 1 and Type 2, respectively. Since the lane width (3 m) is much smaller than the sensing range (160 m), it can be assumed that the SNs in the two-lane scenarios can also obtain environmental data similar to the TaV. However, there are big differences in the perspectives of vehicles in different lanes. For example, in the one-lane scenario, only the horizontal perspectives of the TaV and serving vehicles are different. While in the two-lane scenarios, when a serving vehicle is located in a different lane from the TaV, both the horizontal and vertical perspectives are different. As a result, the computation intensities of coordinate transformation increase compared to that in a one-lane scenario. Considering two-lane scenario, $\alpha_n M_{\text{tra}}$ ($\alpha_n \geq 1$) is introduced to represent the computation intensity of coordinate transformation of the serving vehicle V_n. Note that α_n equals one when V_n is located in the same lane with the TaV, because this is similar to that in the one-lane scenario and they have the same computation intensities of coordinate transformation M_{tra}. When a serving vehicle is located in a different lane from the TaV, α_n equals to 2 when the computation intensity of coordinate transformation in vertical perspective equals to that in horizontal perspective. Without loss of generality, it is assumed that the computation intensity of coordinate transformation in vertical perspective is less than or equal to that in horizontal perspective, i.e., $1 < \alpha_n \leq 2$. Moreover, the computation intensities of coordinate transformation at the RSU in Type 1 and Type 2 are assumed to be the same as that in a one-lane scenario, because the lane width is much smaller than the height of the RSU (tens to hundreds of meters) and there are little differences in the perspectives in different scenarios. It can be seen from Fig. 10 that the ORT performance of TATO always outperforms that of SBO in the two-lane

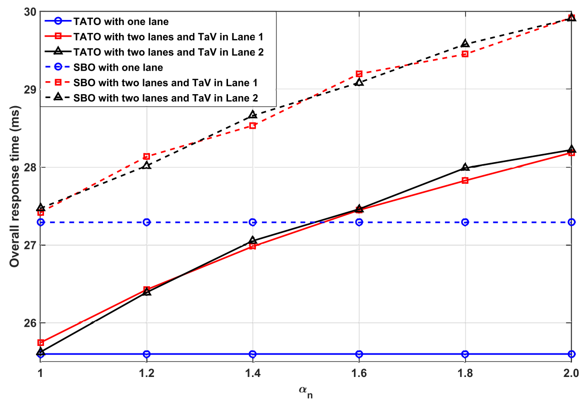


Fig. 10. ORT performance in one-lane and two-lane scenarios.

scenario, which further demonstrates the effectiveness of the proposed TATO mechanism. In addition, the ORTs in Type 1 and Type 2 are similar. The reason is that there are two serving vehicles in different lanes from the TaV in both Type 1 and Type 2, resulting in similar subtasks response time on serving vehicles. Finally, the two-lane scenario always obtains worse ORT performance compared to the one-lane scenario due to the increase of computation intensities of coordinate transformation of the serving vehicles which are located in a different lane from the TaV.

Furthermore, considering a more complicated scenario with many TaVs and many RSUs, it should be noted that the sensors of RSUs can also sense similar environmental data to that of TaVs within its coverage, as long as the deployment of RSUs is dense enough, i.e., the coverage ranges of RSUs are smaller than the sensing ranges of sensors. Moreover, a serving vehicle can sense the environmental data similar to its neighboring TaVs. Therefore, as long as the serving nodes and their resources are reasonably allocated to TaVs, the sensed environmental data of the serving nodes can also completely replace that of the TaVs. Nevertheless, the resource allocation problem may become more complicated and will be investigated in the future.

VI. Conclusion

In this article, we studied the task offloading based on the convergence of communication and sensing in vehicular edge networks. We proposed a new TATO mechanism which allows for the flexible choice of the input data uploading and computation instruction transmission adaptive to the status of the network. We proposed to jointly optimize the task and wireless bandwidth ratios as well as the transmission choices of all nodes to minimize the ORT. By exploiting the special structure, we developed an efficient binary search and feasibility check algorithm for solving the ORT minimization problem. Numerical results demonstrated that the proposed mechanism outperforms two benchmark mechanisms in terms of the ORT at all parameter settings. Furthermore, simulation results showed the applicability of our proposed mechanism to the vehicular applications with heterogeneous requirements. For future work, we are going to consider the impact of the SNs' sensing range on the task ratios.

References

[1] K. Liu, X. Xu, M. Chen, B. Liu, L. Wu, and V. C. S. Lee, "A hierarchical architecture for the future Internet of Vehicles," *IEEE Commun. Mag.*, vol. 57, no. 7, pp. 41–47, Jul. 2019.

[2] B. Liu, D. Jia, J. Wang, K. Lu, and L. Wu, "Cloud-assisted safety message dissemination in VANET-cellular heterogeneous wireless network," *IEEE Syst. J.*, vol. 11, no. 1, pp. 128–139, Mar. 2017.

[3] Q. Ma, Y.-F. Liu, and J. Huang, "Time and location aware mobile data pricing," *IEEE Trans. Mobile Comput.*, vol. 15, no. 10, pp. 2599–2613, Oct. 2016.

[4] Intel. *Technology and Computing Requirements for Self-Driving Cars.* Accessed: Dec. 1, 2017. [Online]. Available: https://www.intel.com/content/www/us/en/automotive/driving-safety-advanced-driver-assistance-systems-self-driving-technology-paper.html

[5] Y. Qi, L. Tian, Y. Zhou, and J. Yuan, "Mobile edge computing-assisted admission control in vehicular networks," *IEEE Veh. Technol. Mag.*, vol. 14, no. 1, pp. 37–44, Mar. 2019.

[6] Y. Zhou *et al.*, "Service aware 6G: An intelligent and open network based on convergence of communication, computing and caching," *Digit. Commun. Netw.*, vol. 6, no. 3, pp. 253–260, Aug. 2020.

[7] Y. Zhou, L. Tian, L. Liu, and Y. Qi, "Fog computing enabled future mobile communication networks—A convergence of communication and computing," *IEEE Commun. Mag.*, vol. 57, no. 5, pp. 20–27, May 2019.

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

[9] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 36–44, Jun. 2017.

[10] J. Wang, D. Feng, S. Zhang, J. Tang, and T. Q. S. Quek, "Computation offloading for mobile edge computing enabled vehicular networks," *IEEE Access*, vol. 7, pp. 62624–62632, 2019.

[11] 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 Commun. Mag.*, vol. 55, no. 7, pp. 94–100, Jul. 2017.

[12] C. Yang, Y. Liu, X. Chen, W. Zhong, and S. Xie, "Efficient mobility-aware task offloading for vehicular edge computing networks," *IEEE Access*, vol. 7, pp. 26652–26664, 2019.

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

[14] Y. Qi, L. Tian, Y. Zhou, J. Yuan, J. Shi, and X. Ge, "MEC-assisted admission control based on convergence of communication and computation," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, 2019, pp. 1–6.

[15] H. Zhou, N. Cheng, J. Wang, J. Chen, Q. Yu, and X. Shen, "Toward dynamic link utilization for efficient vehicular edge content distribution," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8301–8313, Sep. 2019.

[16] M. Sookhak *et al.*, "Fog vehicular computing: Augmentation of fog computing using vehicular cloud computing," *IEEE Veh. Technol. Mag.*, vol. 12, no. 3, pp. 55–64, Sep. 2017.

[17] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3860–3873, Jun. 2016.

[18] F. Malandrino, C. Casetti, C. F. Chiasserini, C. Sommer, and F. Dressler, "The role of parked cars in content downloading for vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 9, pp. 4606–4617, Nov. 2014.

[19] C. Huang, R. Lu, and K.-K. R. Choo, "Vehicular fog computing: Architecture, use case, and security and forensic challenges," *IEEE Commun. Mag.*, vol. 55, no. 11, pp. 105–111, Nov. 2017.

[20] Y. Xiao and C. Zhu, "Vehicular fog computing: Vision and challenges," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, 2017, pp. 6–9.

[21] D. V. Le and C.-K. Tham, "Quality of service aware computation offloading in an ad-hoc mobile cloud," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8890–8904, Sep. 2018.

[22] J. Zhou, F. Wu, K. Zhang, Y. Mao, and S. Leng, "Joint optimization of offloading and resource allocation in vehicular networks with mobile edge computing," in *Proc. 10th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Hangzhou, China, 2018, pp. 1–6.

[23] H. Wang, X. Li, H. Ji, and H. Zhang, "Federated offloading scheme to minimize latency in MEC-enabled vehicular networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Abu Dhabi, United Arab Emirates, 2018, pp. 1–6.

[24] B. Gu, Y. Chen, H. Liao, Z. Zhou, and D. Zhang, "A distributed and context-aware task assignment mechanism for collaborative mobile edge computing," *Sensors*, vol. 18, no. 8, pp. 2423–2439, Aug. 2018.

[25] C. Zhou *et al.*, "Deep reinforcement learning for delay-oriented IoT task scheduling in SAGIN," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 911–925, Feb. 2021.

[26] T. Ye, X. Lin, J. Wu, G. Li, and J. Li, "Toward dynamic computation offloading for data processing in vehicular fog based F-RAN," in *Proc. IEEE 4th Int. Conf. Data Sci. Cyberspace (DSC)*, Hangzhou, China, 2019, pp. 196–201.

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

[28] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid, "Dense reconstruction using 3D object shape priors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Portland, OR, USA, 2013, pp. 1288–1295.

[29] W. Zhuang, Q. Ye, F. Lyu, N. Cheng, and J. Ren, "SDN/NFV-empowered future IoV with enhanced communication, computing, and caching," *Proc. IEEE*, vol. 108, no. 2, pp. 274–291, Feb. 2020.

[30] "Study on enhancement of 3GPP support for 5G V2X services, v15.1.0," 3GPP, Sophia Antipolis, France, Rep. 22.886, Mar. 2017.

[31] S. Zhang, J. Chen, F. Lyu, N. Cheng, W. Shi, and X. Shen, "Vehicular communication networks in automated driving era," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 26–32, Sep. 2018.

[32] G. Liu, W. Wang, J. Yuan, X. Liu, and Q. Feng, "Elimination of accumulated error of 3D target location based on dual-view reconstruction," in *Proc. 2nd Int. Symp. Electron. Commer. Security*, Nanchang, China, 2009, pp. 121–124.

[33] I. Sorkhoh, D. Ebrahimi, R. Atallah, and C. Assi, "Workload scheduling in vehicular networks with edge cloud capabilities," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8472–8486, Sep. 2019.

[34] J. Sun, Q. Gu, T. Zheng, P. Dong, A. Valera, and Y. Qin, "Joint optimization of computation offloading and task scheduling in vehicular edge computing networks," *IEEE Access*, vol. 8, pp. 10466–10477, 2020.

[35] Y. Sun *et al.*, "Adaptive learning-based task offloading for vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3061–3074, Apr. 2019.

[36] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE Trans. Commun.*, vol. 64, no. 10, pp. 4268–4282, Oct. 2016.

[37] H. Ogata and T. Takahashi, "A geometric approach to task understanding and playback: Compact and robust task description for complex environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1. San Diego, CA, USA, 1994, pp. 848–854.

[38] M. Abdulla, E. Steinmetz, and H. Wymeersch, "Vehicle-to-vehicle communications with urban intersection path loss models," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Washington, DC, USA, 2016, pp. 1–6.

[39] M. Hong, Q. Li, and Y.-F. Liu, "Decomposition by successive convex approximation: A unifying approach for linear transceiver design in heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 2, pp. 1377–1392, Feb. 2016.

[40] J. Nocedal and S. J. Wright, "Numerical optimization," 2nd ed. New York, NY, USA: Springer, 2006.

[41] L. Condat, "Fast projection onto the simplex and the l1 ball," *Math. Program.*, vol. 158, pp. 575–585, 2016.

[42] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 12, pp. 2510–2523, Dec. 2015.

[43] *User Equipment (UE) Radio Transmission and Reception*, 3GPP Standard TS 38.101, 2021.

[44] P. Liu, J. Li, and Z. Sun, "Matching-based task offloading for vehicular edge computing," *IEEE Access*, vol. 7, pp. 27628–27640, 2019.

[45] Y. Liu, S. Wang, J. Huang, and F. Yang, "A computation offloading algorithm based on game theory for vehicular edge networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[46] J. Lee and T. Park, "Fast Lidar-camera fusion for road detection by CNN and spherical coordinate transformation," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Paris, France, 2019, pp. 1797–1802.

[47] M. Teichmann, M. Weber, M. Zöllner, R. Cipolla, and R. Urtasun, "MultiNet: Real-time joint semantic reasoning for autonomous driving," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Changshu, China, 2018, pp. 1013–1020.

[48] L. Xiao, R. Wang, B. Dai, Y. Fang, D. Liu, and T. Wu, "Hybrid conditional random field based camera-Lidar fusion for road detection," *Inf. Sci.*, vol. 432, pp. 543–558, Mar. 2018.

Yanli Qi received the B.S. degree in communication engineering and the M.S. degree in electronics and communication engineering from Jiangxi University of Science and Technology, Ganzhou, China, in 2014 and 2017, respectively. She is currently pursuing the Ph.D. degree in technology of computer application with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China.

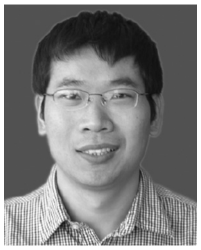
Her research focuses on mobile edge computing, the convergence of communication, computation and caching, and resource management.

Ms. Qi has also served as a reviewer for a number of referred journals and international conferences.

Yiqing Zhou (Senior Member, IEEE) received the B.S. degree in communication and information engineering and the M.S. degree in signal and information processing from Southeast University, Nanjing, China, in 1997 and 2000, respectively, and the Ph.D. degree in electrical and electronic engineering from the University of Hong Kong, Hong Kong, in 2004.

She is currently a Professor with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. She has published over 150 articles and four books/book chapters in the areas of wireless mobile communications.

Prof. Zhou received best paper awards from WCSP2019, IEEE ICC2018, ISCIT2016, PIMRC2015, ICCS2014, and WCNC2013. She also received the 2014 Top 15 Editor Award from IEEE TVT and the 2016–2017 Top Editors of ETT. She is also the TPC Co-Chair of ChinaCom2012, an Executive Co-Chair of IEEE ICC2019, a Symposia Co-Chair of ICC2015, a Symposium Co-Chair of GLOBECOM2016 and ICC2014, a Tutorial Co-Chair of ICCC2014 and WCNC2013, and the Workshop Co-Chair of SmartGridComm2012 and GlobeCom2011. She is also the Associate/Guest Editor of IEEE INTERNET OF THINGS JOURNAL, IEEE TRANSACTIONS VEHICULAR TECHNOLOGY, IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS (Special issue on "Broadband Wireless Communication for High Speed Vehicles" and "Virtual MIMO"), *Transactions on Emerging Telecommunications Technologies*, and *Journal of Computer Science and Technology*.



Ya-Feng Liu (Senior Member, IEEE) received the B.Sc. degree in applied mathematics from Xidian University, Xi'an, China, in 2007, and the Ph.D. degree in computational mathematics from Chinese Academy of Sciences (CAS), Beijing, China, in 2012.

During his Ph.D. study, he was supported by the Academy of Mathematics and Systems Science (AMSS), CAS, to visit Prof. Z.-Q. (Tom) Luo with the University of Minnesota (Twins Cities), Minneapolis, MN, USA, from 2011 to 2012. After his graduation, he joined the Institute of Computational Mathematics and Scientific/Engineering Computing, AMSS, CAS, Beijing, in 2012, where he became an Associate Professor in 2018. His main research interests are nonlinear optimization and its applications to signal processing, wireless communications, and machine learning.

Dr. Liu received the Best Paper Award from the IEEE International Conference on Communications in 2011, the Best Student Paper Award from the International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt) in 2015, the Chen Jingrun Star Award from the AMSS in 2018, the Science and Technology Award for Young Scholars from the Operations Research Society of China in 2018, and the 15th IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award in 2020. He currently serves an Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and an Associate Editor for the IEEE SIGNAL PROCESSING LETTERS and the *Journal of Global Optimization*. He is an Elected Member of the Signal Processing for Communications and Networking Technical Committee of the IEEE Signal Processing Society.

Ling Liu (Member, IEEE) received the B.S. degree in communication engineering from Nanchang University, Nanchang, China, in 2012, and the Ph.D. degree in computer science and technology from the University of Chinese Academy of Sciences, Beijing, China, in 2018.

She is currently an Assistant Professor with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing. Her research focuses on interference and resource management in ultradense networks, and the convergence of communication, computing and caching.

Dr. Liu received the Best Paper Award from IEEE ICC2018. She has also served as a reviewer for a number of referred journals and international conferences.

Zhengang Pan (Senior Member, IEEE) received the Ph.D. degree from the Department of Electrical and Electronic Engineering, University of Hong Kong, Hong Kong, in 2004.

He is the VP of the Advanced Technology Research Institute of Unisoc, where he is currently leading a team working on the research, standardization, prototyping of key technologies for 5G and beyond (B5G/6G). Before joining Unisoc, he was a Principle Staff of China Mobile Research Institute, where he ran the 5G research team for more than three years. He has also been working with Hong Kong ASTRI for more than six years, where he has been involved in multiple technical fields, from wireless communication (WiFi, WiMax, and LTE), to mobile digital TV (T-DMB, DVB-T/H, and CMMB), to wireline broadband access (HomePlug and MoCA), in both system/algorithm design and terminal SoC chip implementation. He has also been working with NTT DoCoMo Beijing Communication Labs Co., Ltd, on the frontier research for 4G wireless communication standards, including 802.11n, 802.16d/e, HSPA, and LTE. He has expertise in many technical fields, including time/frequency/sampling synchronization technology for single-carrier/ multicarrier(OFDM/A)-based system, channel estimation, forward error correction coding, multiple antennas systems (MIMO), and space-time processing/coding and cross layer optimization. He has published more than 100 papers in top journals and international conferences, and filed more than 100 patents with more than 80 grant.

Dr. Pan was also the Vice-Chair of Technical WG of China IMT-2020 PG.